



KINERJA PROTOKOL GREEDY PERIMETER STATELESS ROUTING PROTOKOL (GPSR) MENGGUNAKAN ROADSIDE UNIT(RSU) PADA VEHICLE AD-HOC NETWORK(VANET) DI AREA KOTA MALANG

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:

Ragil Setyo Utomo

NIM: 1415150207111024



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

MALANG

2020



PENGESAHAN

KINERJA PROTOKOL GREEDY PERIMETER STATELESS ROUTING PROTOKOL (GPSR)
MENGUNAKAN ROADSIDE UNIT (RSU) PADA VEHICLE AD-HOC NETWORK
(VANET) DI AREA KOTA MALANG

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

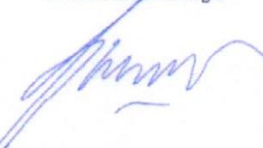
Disusun Oleh :
Ragil Setyo Utomo
NIM: 145150207111024

Skripsi ini telah diuji dan dinyatakan lulus pada
26 Juni 2020

Dosen Pembimbing I


Ir. Primantara Hari Trisnawan, M. Sc.
NIP. 19680912 199403 1 002

Dosen Pembimbing 2


Reza AndriaSiregar, S.T., M.Kom.
NIP. 201102 860805 1 001

Mengetahui



Kota Malang Jurusan Teknik Informatika


Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP. 19710518 200312 1 001



PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 5 Juni 2020



Ragil Setyo Utomo

NIM: 145150207111024



ABSTRAK

Ragil Setyo Utomo, KINERJA PROTOKOL GREEDY PERIMETER STATELESS ROUTING PROTOKOL (GPSR) MENGGUNAKAN ROADSIDE UNIT(RSU) PADA VEHICLE AD-HOC NETWORK(VANET) DI AREA KOTA MALANG

Pembimbing: Ir. Primantara Hari Trisnawan, M.Sc. dan Reza Andria Siregar, S.T., M.Kom.

Vehicular ad-hoc network (VANET) merupakan teknologi jaringan wireless yang memungkinkan kendaraan dapat berkomunikasi ke kendaraan lain. Vanet memiliki 3 cara dalam berkomunikasi, yaitu komunikasi kendaraan dengan kendaraan (V2V), infrastruktur dengan kendaraan (I2V) atau (V2I), dan infrastruktur dengan infrastruktur (I2I). Penelitian ini berfokus terhadap jarak peneletakan tiap Road Side Unit (RSU). Dalam komunikasi yang terjadi di VANET, diperlukan performa yang baik dalam pengiriman data dari kendaraan ke kendaraan maupun kendaraan ke RSU ataupun sebaliknya agar komunikasi antar kendaraan dan juga RSU dapat tersampaikan lebih cepat dan presentase keberhasilan tersampainya suatu informasi akan bertambah. Pada penelitian ini akan diimplementasikan protokol Greedy Perimeter Stateless Routing (GPSR) dengan RSU untuk mengetahui jarak tiap RSU berapakah yang lebih baik untuk diimplementasikan. Implementasi dilakukan menggunakan Network Simulator (NS-2) dan SUMO. Hasil dari Implementasi akan dianalisis dengan parameter uji yang berupa throughput, end-to-end delay dan packet delivery ratio. Berdasarkan hasil pengujian berdasarkan kepadatan kendaraan, RSU dengan jarak tiap 300 m memiliki nilai throughput dan packet delivery ratio yang lebih baik yaitu 1570 bps dan 85,7143% dibandingkan dengan nilai *throughput* RSU dengan jarak tiap 400 m yaitu 1530 bps dan 500 m dengan nilai 1520 bps dan juga hasil *packet delivery ratio* RSU tiap 400 m adalah 77,7778% dan nilai tiap 500 m adalah 75%. Akan tetapi pada hasil implementasi berdasarkan kecepatan, Untuk nilai end-to-end delay serta packet delivery ratio hasil terbaik didapatkan oleh skenario 60km/jam dengan jarak RSU 300m dan 400m dengan nilai yang sama yaitu 6,26243 m/s untuk niai end-to-end delay dan 92,8571% untuk nilai packet delivery ratio. Dengan kata lain, peneletakkan RSU berdasarkan jarak berpengaruh terhadap kinerja routing.

Kata kunci: VANET, NS-2, SUMO, GPSR, RSU



ABSTRACT

Ragil Setyo Utomo, Performance of Greedy Perimeter Stateless Routing(GPSR) Protocol Using Roadside Unit(RSU) with Vehicle Ad-Hoc Network(VANET) in Malang City Area.

Supervisors: Ir. Primantara Hari Trisnawan, M. Sc. and Reza Andria Siregar, S.T., M.Kom.

Vehicular ad-hoc network (VANET) is a wireless network technology that allow vehivle can communicate with ohter vehicle. VANET have 3 way to communicate, that is vehicle to vehicle communication (V2V), infrastructure to vehicle (I2V) or (V2I), and infrastructure to infrastructure (I2I). This study focused at the distance placement in each road side unit (RSU). Communication in VANET, its need good performance in sending data from vechile to vehicle for communication between vehicle to vehicle as well as vehicle to RSU or the other round can be delivered faster and the percentage of success information arrived is increasing. In this study protocol greedy perimeter stateless routing (GPSR) will be implement with RSU to know which distant in each RSU is better to implement. Implementation will be worked with Network Simulator (NS-2) and SUMO. The result from this implementation will be analyzed with test parameters that is throughput, end-to-end delay, and packet delivery ratio. From the result based on vehicle density, RSU with distant 300 m have better value, that is 1570 bps and 85,7143% compared from RSU with distance 400 m, that is 1530 bps and 500m with value 1520 bps. And also the value of packet delivery ratio for RSU with 400 m is 77,7778% and 500 m with 75%. However on implemention result based on vehicle speed, the best value in end-to-end delay and packet delivery ratio gotten by scenario of 60 km/hr with RSU 300m and 400m with the same value that is 6,26243 m/s for the end-to-end-delay and 92,8571% for packet delivery ratio. In other word, RSU placement with distance is taking effect toward routing performance.

Keywords: VANET, NS-2, SUMO, GPSR, RSU



DAFTAR ISI

DAFTAR ISI	ii
DAFTAR GAMBAR	iii
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Identifikasi Masalah	3
1.3 Rumusan Masalah	3
1.4 Tujuan	4
1.5 Manfaat	4
1.6 Batasan Masalah	4
1.7 Sistematika Pembahasan	5
BAB 2 LANDASAN KEPUSTAKAAN	6
2.1 Kajian Pustaka	6
2.1 Vehicular ad hoc Network (VANET)	7
2.2 Roadside Unit (RSU)	8
2.3 Greedy Perimeter Stateless Routing (GPRS)	10
2.4 Network Simulator (NS2)	11
2.5 Simulation of Mobility (SUMO)	12
2.6 Parameter Pengujian	12
2.6.1 Throughput	12
2.6.2 End to End Delay	12
2.6.3 Packet delivery ratio (PDR)	13
BAB 3 METODOLOGI	14
3.1 Studi Literatur	14
3.2 Perancangan Sistem	15
3.2.1 Gambaran Umum Sistem	15
3.2.1 Perancangan Skenario	16
3.2.1 Perancangan <i>Greedy Perimeter Stateless Routing</i> (GPSR)	17
3.2.1 Perancangan Peletakan <i>Roadside Unit</i> (RSU)	18
3.2.1 Perancangan Simulasi di <i>Network Simulator</i> (NS-2)	20
3.3 Implementasi	21



3.4 Pengujian dan Analisis	21
3.4.1 Pengujian Terhadap Kepadatan Kendaraan.....	22
3.4.1 Pengujian Terhadap Kecepatan Kendaraan.....	23
3.5 Kesimpulan dan Saran	24
BAB 4 IMPLEMENTASI	25
4.1 Implementasi Skenario	25
4.2 Implementasi Road Side Unit(RSU)	26
4.3 Implementasi Simulasi NS2.....	27
4.4 Implementasi Pengujian	29
4.4.1 Implementasi Throughput	30
4.4.2 Implementasi End-to-end Delay	30
4.4.1 Implementasi Packet Delivery Ratio.....	31
BAB 5 PENGUJIAN DAN ANALISIS.....	32
5.1 Pengujian Terhadap Kepadatan Kendaraan.....	32
5.1.1 Throughput.....	32
5.1.2 End-to-end Delay.....	34
5.1.3 Packet Delivery Ratio	36
5.2 Pengujian Skenario Terhadap Kecepatan Kendaraan.....	38
5.2.1 Throughput.....	39
5.2.2 End-to-end Delay.....	41
5.2.3 Packet Delivery Ratio	43
BAB 6 PENUTUP	45
6.1 Kesimpulan.....	45
6.2 Saran	45
DAFTAR PUSTAKA.....	47
LAMPIRAN	49



DAFTAR TABEL

Tabel 2.1 Kajian Pustaka	6
Tabel 3.1 Parameter Simulasi.....	20
Tabel 3.2 Skenario Pengujian	22
Tabel 4.1 Perintah mengkonversi file OSM.....	25
Tabel 4.2 Perintah membuat node dengan tools randomTrips.py.....	25
Tabel 4.3 Perintah membuat file map.sumo.cfg.....	25
Tabel 4.4 Konfigurasi file map.sumo.cfg.....	26
Tabel 4.5 Perintah menjalankan map.sumo.cfg pada SUMO	26
Tabel 4.6 Perintah mengkonversi mengkonversi map.sumo.cfg ke skenario.xml	26
Tabel 4.7 Perintah mengkonversi(.xml) ke (.tcl)	26
Tabel 4.8 Implementasi Node ke RSU.....	27
Tabel 4.9 Konfigurasi wireless-gpsr.tcl.....	28
Tabel 4.10 Konfigurasi cbr100.tcl.....	28
Tabel 4.11 Konfigurasi mobility.tcl.....	29
Tabel 4.12 Perintah menjalankan script.tcl pada <i>wireless-gpsr</i>	29
Tabel 4.13 Hasil trace file	30
Tabel 4.14 Perintah menjalankan script.awk pada throughput	30
Tabel 4.15 Perintah menjalankan script.awk pada End-to-end delay.....	30
Tabel 4.16 Perintah menjalankan Script.awk Packet Delivery Ratio.....	31
Tabel 5.1 Hasil Pengujian nilai Throughput	32
Tabel 5.2 Hasil Pengujian nilai end-to-end delay.....	34
Tabel 5.3 Hasil Pengujian nilai packet delivery ratio	36
Tabel 5.4 Hasil Pengujian nilai throughput.....	39
Tabel 5.5 Hasil pengujian nilai end-to-end delay.....	41
Tabel 5.6 Hasil pengujian nilai packet delivery ratio	43



DAFTAR GAMBAR

Gambar 2.1 Jaringan <i>Vehicular adhoc</i> dan beberapa kemungkinan aplikasi	7
Gambar 2.2 RSU melebarkan jarak jaringan ad hoc dengan menerukan data OBU	9
Gambar 2.3 RSU bekerja sebagai sumber informasi	9
Gambar 2.4 RSU memberikan pelayanan internet ke OBU lain	9
Gambar 2.5 Metode <i>greedy forwarding</i>	10
Gambar 2.6 Metode <i>perimeter forwarding</i>	11
Gambar 2.7 Arsitektur NS 2	11
Gambar 3.1 Diagram Rancangan Simulasi	15
Gambar 3.2 Diagram Alur Perancangan Peta Skenario	16
Gambar 3.3 Flowchart Protokol GPSR	17
Gambar 3.4 Gambar <i>Openstreet Map</i>	18
Gambar 3.5 Gambar Pengambilan Letak Jarak RSU	18
Gambar 3.6 Gambar Topologi Peneletakan RSU Tiap 300m	19
Gambar 3.7 Gambar Topologi Peneletakan RSU Tiap 400m	19
Gambar 3.8 Gambar Topologi Peneletakan RSU Tiap 500m	20
Gambar 3.9 Flowchart Perancangan Pengujian Berdasarkan Banyak Kendaraaan	22
Gambar 3.10 Flowchart Perancangan Pengujian Berdasarkan Kecepatan Kendaraaan	23
Gambar 6.1 Peletakan RSU di Simulator <i>Nam</i>	32
Gambar 6.2 Grafik <i>throughput</i> pengujian kepadatan kendaraan	33
Gambar 6.3 Trace File <i>Throughput</i> Kendaraan 180 dengan RSU 300m	34
Gambar 6.4 Trace File <i>Throughput</i> Kendaraan 180 Tanpa RSU	34
Gambar 6.5 Grafik <i>end-to-end delay</i> pengujian kepadatan kendaraan	35
Gambar 6.6 Trace File <i>end to end delay</i> Kendaraan 180 RSU 300m	35
Gambar 6.7 Trace File <i>end to end delay</i> Kendaraan 180 RSU 500m	36
Gambar 6.8 Grafik <i>packet delivery ratio</i> pengujian kepadatan kendaraan	36
Gambar 6.9 Trace File <i>Packet Delivery Ratio</i> Kendaraan 180 RSU 300m	37
Gambar 6.10 Trace File <i>Packet delivery Ratio</i> Kendaraan 180 Tanpa RSU	38
Gambar 6.11 Grafik <i>Throughput</i> Pengujian Kecepatan Kendaraan	39



Gambar 6.12 Trace File *Throughput* Kecepatan Kendaraan 50km/jam RSU 500m

..... 40

Gambar 6.13 Trace File *Throughput* Kecepatan Kendaraan 50km/jam RSU 300m

..... 40

Gambar 6.14 Grafik *End-to-end delay* Pengujian Kecepatan Kendaraan..... 41

Gambar 6.15 Trace File *end-to-end delay* Kecepatan Kendaraan 60km/jam RSU

300m 42

Gambar 6.16 Trace File *end-to-end delay* Kecepatan Kendaraan 60km/jam RSU

500m 42

Gambar 6.13 Grafik *Packet Delivery Ratio* Pengujian Kecepatan Kendaraan..... 43



BAB 1 PENDAHULUAN

1.1 Latar belakang

Vehicular ad hoc network atau disingkat VANET adalah jaringan pada kendaraan yang digunakan untuk memberikan pelayanan komunikasi antar node kendaraan satu dengan node kendaraan yang lain. Pada VANET, node kendaraan bisa mengorganisir nodenya sendiri dan berkomunikasi kepada node sesama di lingkungan yang tidak menggunakan infrastruktur. VANET dapat mengurangi resiko kecelakaan, mengatur trafik pada lalu lintas kendaraan, memberikan informasi tentang kuota parkir, arah ke pom bensin, dan masih banyak lagi (Abbasi, & Khan, 2018). Penggunaan protokol routing pada VANET harus dilandasi berdasarkan kategori yang sudah ada pada VANET, jika tidak sesuai maka akan menimbulkan permasalahan baru pada jaringan VANET tersebut.

Pada VANET protokol routing dilakukan pembagian hingga terbagi menjadi 5, yaitu *position based protocol*, *geographic based routing protocol*, *topology based routing protocol*, *cluster based routing protocol*, dan *broadcast routing protocol*. Di dalam *geographic based protocol* terdapat pembagian jenis protokol yang dipakai, yaitu *overlay* dan *non overlay*. Pada *position based protocol* juga terdapat beberapa jenis protokol yang dapat dilakukan pengelompokan berdasar kepada arsitektur jaringannya yaitu, *proactive*, *reactive*, dan *hybrid*. Dalam VANET jumlah kendaraan yang padat merupakan suatu masalah dalam memilih protokol yang cocok. Dalam hal ini *geographic based protocol* mempunyai keunggulan yaitu cocok digunakan untuk kepadatan kendaraan yang tinggi daripada jenis *routing protocol* yang lain (Paul, et al., 2011). Salah satu *geographic based routing protocol* yang sedang dikembangkan adalah *Greedy Perimeter Stateless Routing* (GPSR).

Pada kendaraan yang terpasang *On Board Unit* (OBU) yang fungsinya untuk saling tukar menukar informasi, yang diantaranya : kecepatan, posisi, arah node tetangga, dan waktu. Dalam hal ini salah satu protokol yang dapat memaksimalkan penggunaan dari GPS dengan memanfaatkan posisi kendaraan berada adalah *Greedy Perimeter Stateless Routing* (GPSR). Protokol GPSR merupakan salah satu jenis dari protokol yang cara kerjanya menggunakan *geographic based routing protocol*. Jenis protokol ini menggunakan informasi lokasi dari node tetangga terdekat sebagai perantara dalam pengiriman paket. Protokol GPSR akan mengirimkan pesan yang berupa *hello message* secara periodik untuk memperbaharui atau mendapatkan informasi berupa lokasi *node* yang ada dalam jangkauan transmisi. Terdapat 2 metode pada protokol GPSR dalam pengiriman paket data, yaitu *Greedy Forwarding* dan *Perimeter Forwarding*. *Greedy Forwarding* merupakan metode pengiriman paket data dengan mencari dan meneruskan paket ke node tetangga terdekat dengan informasi dari *node* ke *node* lain bisa menemukan langsung ke *node* tujuan. *Perimeter Forwarding* akan aktif jika pada metode *Greedy Forwarding* sedang tidak dapat bekerja atau *node* tidak dapat menemukan node tetangga terdekat yang mempunyai informasi untuk bisa meneruskan paket ke *node* tujuan, sehingga cara *Perimeter Forwarding*



memanfaatkan *node* yang ada di sekitarnya untuk mengirimkan permintaan berupa informasi ke *node* tetangga dan *node* tetangga tersebut meneruskan permintaan tersebut ke *node* tetangga terdekat lainnya, proses pengiriman paket data ini berjalan hingga pada di suatu paket data yang berisikan informasi dari *node-node* yang dilewati tadi dikirimkan ke *node* sumber yang dilakukan oleh *node* terakhir yang menerima permintaan informasi tersebut. *Perimeter forwarding* akan terus dilakukan sampai informasi *node* tujuan didapatkan oleh *node* sumber dan proses ini lebih lama dibandingkan *Greedy Forwarding* (saputra, et al., 2018).

VANET mempunyai beberapa masalah karena adanya karakteristik VANET yang berbeda dari penggunaan komunikasi wireless yang lain. Salah satunya adalah VANET harus bisa menyampaikan informasi ke *node* atau kendaraan lain dengan cepat dan informasi itu tidak hilang di tengah jalan agar kendaraan dapat menerima informasi mengenai jalur yang ingin dilewatinya dan menghindari jalur yang sedang macet (Kumar, & Dave, 2011). Pada kasus ini dibutuhkan sesuatu yang dapat mempercepat sampainya suatu paket data dan meningkatkan keberhasilan rasio pengiriman agar informasi dapat cepat tersampaikan dan tidak hilang.

Roadside Unit (RSU) merupakan jaringan *Vehicle-to-infrastructure* (V2I). Kegunaan infrastruktur adalah menaikkan kemampuan komunikasi yang sangat berguna untuk aplikasi keselamatan, dimana pesan bahaya sampai kepada yang berotoritas, juga kendaraan sekitar yang berefek pada kecelakaan, kesalahan fungsi, dan lainnya. RSU yang salah satu kegunaannya adalah sebagai sumber informasi dan apabila diletakkan di tempat yang efektif, maka akan sangat berguna untuk kendaraan yang membutuhkan informasi dengan cepat (Fogue, et al., 2018).

Pada protokol GPSR terdapat metode yang memakan banyak waktu yaitu *perimeter forwarding* dimana routing dengan metode ini membutuhkan waktu yang lebih lama daripada metode *Greedy Forwarding* dan karena proses pengiriman suatu data sangat bergantung pada *node* kendaraan yang merupakan perantara suatu pengiriman, maka tingkat keberhasilan suatu paket data pun bervariasi. Untuk itu diperlukan RSU sebagai *node* tambahan yang tidak bergerak yang berguna sebagai perantara penerus paket data supaya meningkatkan keberhasilan suatu paket data bisa tersampaikan sampai ke tujuan dan mempercepat laju kecepatan jaringan yang dipakai. Dengan adanya RSU, maka jenis komunikasi di VANET dapat bervariasi yang berupa *vehicle to vehicle* (V2V), *Vehicle to Infrastructure* (V2I), dan *Infrastructure to vehicle* (I2V) dimana RSU dapat menerima paket data dan meneruskannya ke kendaraan lain yang berada dalam jalur dan jangkauannya.

Berdasarkan penelitian Abdradou, dan Zhuang (2011) yang termasuk dalam melakukan analisis peneletakkan RSU dengan jumlah minimal untuk melingkupi ruas jalan, digunakan skenario berdasarkan jarak antar RSU yang akan diletakkan dan mendapatkan hasil rata-rata *throughput* yang lebih tinggi dibandingkan



dengan tidak memakai RSU. Untuk itu dilakukanlah penelitian berdasarkan jarak antar RSU karena akan meningkatkan performa seperti *throughput*.

Kota Malang merupakan salah satu kota terbesar di Jawa Timur yang mempunyai kepadatan kendaraan yang sangat padat dengan mobilitas kendaraan yang tinggi, hal ini dapat menyebabkan banyak kendaraan yang tidak mempunyai informasi jalan akan terjebak dalam kemacetan. Berdasarkan hal ini diperlukanlah VANET supaya pengguna kendaraan dapat memiliki informasi tentang keadaan jalur yang akan dilewati serta pemilihan rute alternatif jika jalur utama sedang padat dan informasi lainnya dengan baik dan juga cepat.

Pada penelitian ini permasalahan yang ada adalah Kota Malang memerlukan jaringan VANET yang dapat mengirim pesan dengan cepat dan presentase keberhasilan yang baik agar kendaraan agar pengendara mendapatkan informasi yang diinginkan dengan cepat supaya kendaraan tersebut dapat mengambil jalur alternatif sehingga dapat menghindari jalan yang macet. Untuk itu penulis melakukan penelitian tentang Kinerja protokol *Greed Perimeter Stateless Routing* (GPSR) menggunakan *Roadside Unit* (RSU) pada *Vehicle Ad-hoc Network* (VANET) di area kota Malang. Penelitian ini menggunakan protokol GPSR karena protokol tersebut menggunakan penerusan data secara geografis serta menggunakan jalan terpendek untuk menemui rutenya. Simulasi dilakukan dengan menggunakan NS-2 serta simulator SUMO yang dapat berperan sebagai *traffic simulator*. Untuk parameter yang digunakan pada penelitian ini adalah *throughput*, *packet delivery ratio*, dan *end to end delay*.

1.2 Identifikasi Masalah

Berdasarkan uraian latar belakang tersebut, maka identifikasi masalahnya adalah sebagai berikut:

1. Kendaraan kota Malang membutuhkan penyampaian informasi dengan cepat.
2. Vanet membutuhkan protokol routing yang sesuai dengan topologi yang dipakai.

Berdasarkan identifikasi masalah di atas, maka penulis mengusulkan penggunaan Vanet dengan menggunakan RSU agar informasi dapat disampaikan dengan cepat dan baik. Serta penggunaan protokol routing yang digunakan adalah GPSR karena GPSR adalah merupakan salah satu *geographic based routing* yang cocok digunakan pada suatu wilayah perkotaan.

1.3 Rumusan masalah

Berdasarkan uraian latar belakang tersebut, maka dirumuskan masalah sebagai berikut:

3. Bagaimana implementasi protokol routing GPSR pada VANET dengan menggunakan RSU pada komunikasi kendaraan menggunakan simulator NS-2?
4. Bagaimana hasil parameter *Quality of Services* (QoS) dari penerapan simulasi protokol GPSR menggunakan RSU berdasarkan jarak yang ditentukan pada VANET untuk komunikasi dengan kendaraan?



1.4 Tujuan

Tujuan dari penelitian ini adalah untuk mengetahui hasil kinerja dari protokol GPSR pada VANET yang menggunakan RSU pada simulasi berdasarkan daerah Soekarno Hatta dan sekitarnya di kota Malang. Dengan hasil kinerja yang didapat, maka didapatkan seberapa baik hasil kinerja RSU berdasarkan jarak yang sudah ditentukan menggunakan protokol GPSR.

1.5 Manfaat

- Bagi Penulis :

1. Mendapatkan pengetahuan tentang VANET.
2. Sebagai referensi untuk melakukan penelitian tugas akhir dengan tema yang sama.

- Bagi pembaca :

1. Dapat memahami bagaimana kinerja dari GPSR dengan memakai RSU.
2. Mendapatkan pengetahuan protokol GPSR serta node pembantu yang bernama RSU.

1.6 Batasan masalah

Adapun batasan masalah dalam penelitian ini sebagai berikut :

1. Komunikasi kendaraan yang dipakai di penelitian ini adalah komunikasi kendaraan menggunakan RSU.
2. RSU diletakkan tiap 300 m, 400 m, dan 500 m di sepanjang jalan raya.
3. Protokol yang digunakan merupakan protokol GPSR (*Greedy Perimeter Stateless Routing*).
4. *Operating system* yang dipakai adalah Linux.
5. Protokol routing yang digunakan GPSR.
6. Area simulasi diambil di daerah Soekarno Hatta dan sekitarnya pada kota Malang.
7. Parameter penelitian yang diukur adalah *throughput*, *packet delivery ratio*, dan *end-to-end delay*.
8. Simulasi protokol dilakukan dengan memakai simulator NS-2
9. Simulasi kendaraan pada lalu lintas dilakukan menggunakan simulator SUMO.



1.7 Sistematika pembahasan

Sistematika penulisan laporan ditunjukkan untuk memberikan gambaran dan uraian dari laporan penelitian secara garis besar yang meliputi beberapa bab, sebagai berikut:

BAB I Pendahuluan

Bab ini berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, sistematika penulisan laporan dari *Studi Penempatan Roadside Unit (RSU) Pada Vehicle Ad-Hoc Network (VANET) di Area Kota Malang*.

BAB II Landasan Kepustakaan

Bab ini menguraikan kajian pustaka dan referensi yang mendasari penelitian yang dilakukan.

BAB III Metodologi

Bab ini akan dijelaskan tentang metode yang nantinya akan digunakan untuk melakukan penelitian secara garis besar.

BAB IV Perancangan dan Implementasi

Bab ini menguraikan gambaran dari perancangan sistem yang diteliti yang nantinya akan dibuat laporan dari implementasi.

BAB V Pengujian dan Analisis

Bab ini akan dilakukan pengujian dan analisis agar mengetahui kinerja yang sudah diterapkan pada tahap perancangan dan implementasi.

BAB VI Penutup

Menguraikan kesimpulan yang diperoleh dari hasil penelitian yang telah dilakukan serta saran-saran untuk pengembangan lebih lanjut.



BAB 2 LANDASAN KEPUSTAKAAN

Teori serta konsep yang mendasari penelitian yang dilakukan adalah dengan adanya penelitian sebelumnya dari penulis lain yang disebarakan atau publikasikan berupa jurnal atau paper.

2.1 Kajian Pustaka

Pada tahap kajian pustaka di penelitian ini berisi tentang bahasan penelitian sebelumnya yang menunjang penelitian yang diusulkan. Kajian Pustaka ini didasari oleh penelitian sebelumnya yang terkait pada bahasan VANET serta protokol routing GPSR.

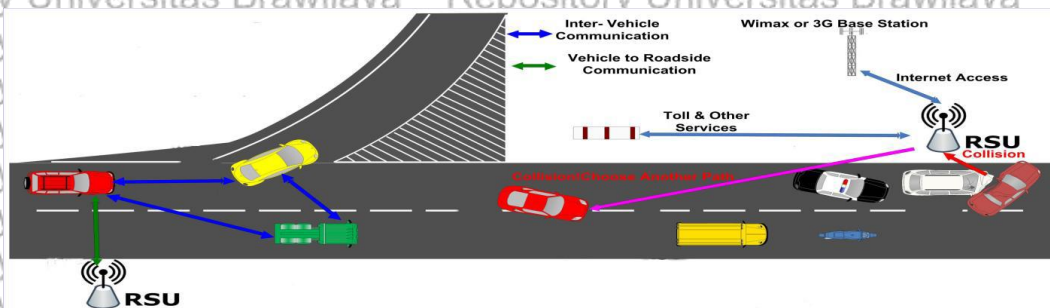
No	Nama peneliti Tahun dan Judul	Persamaan	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
1.	A. R. Deshmukh, dan S. S. Dorle 2016, <i>Simulasi of Urban Mobility (Sumo) For Evaluating Qos Parmeters For Vehicular Adhoc Network.</i>	Analisis performansi pada VANET menggunakan an simulasi Sumo	Menganalisis performansi VANET menggunakan protokol AODV dan AMODV pada Vanet menggunakan Sumo dan <i>Java Open Street Map Editor</i> (JOSM).	Menganalisis performansi VANET menggunakan protokol GPSR dengan menggunakan simulator NS-2.
2.	Faisal H. Saputra, Radityo Anggoro, dan Supeno Djanali 2018, STUDI KINERJA GREEDY PERIMETER STATELESS ROUTING BERBASIS OVERLAY NETWORK PADA VANET.	Analisis performansi VANET menggunakan an protokol GPSR.	Menganalisis Performansi VANET dengan membandingkan protokol GPSR dengan DSR.	Menganalisis Performansi VANET menggunakan RSU dengan protokol GPSR.

Tabel 2.1 Kajian Pustaka



2.2 Vehicular ad-hoc Network (VANET)

Vehicular ad-hoc Network (VANET) merupakan teknologi jaringan wireless ad-hoc yang memungkinkan kendaraan dapat berkomunikasi ke kendaraan lain (V2V) atau infrastruktur (V2I). Tujuan VANET dibuat agar pengendara dapat berkendara dengan aman dan nyaman. Komunikasi yang dilakukan pada kendaraan ke kendaraan lain akan mengakibatkan antar kendaraan untuk bertukar informasi seperti : jarak antar kendaraan, tempat pom bensin, kemacetan jalan serta kecepatan maksimal kendaraan lain (Goyal, & Soni, 2014). VANET bekerja dengan membangun jaringan wireless yang mengirimkan informasi ke kendaraan yang membutuhkan melalui kendaraan maupun infrastruktur yang ada. Dengan menggunakan *On Board Unit* (OBU) yang fungsinya untuk menulis dan membaca serta menampung informasi, pertukaran informasi pun dapat dilakukan. Gambar 2.1 menunjukkan bentuk beberapa aplikasi VANET seperti *vehicle collision warning*, serta *security distance warning* dan menunjukkan bagaimana VANET dapat bekerja yaitu bisa melalui kendaraan ataupun melalui RSU.



Gambar 2.1 Jaringan *Vehicular adhoc* dan beberapa kemungkinan aplikasi

Sumber : (Paul, et al.,2011)

VANET mempunyai karakteristik yang unik dan membuatnya berbeda dari MANET (Saini, & Singh, 2015), diantaranya adalah :

1. *High Mobility*

VANET didesain agar dapat mengatasi mobilitas jaringan yang tinggi karena *Node* yang ada di VANET bisa bergerak dalam kecepatan tinggi.

2. *Network Topology*

Karena tingginya mobilitas node dan kendaraan bergerak dengan kecepatan yang berubah-ubah, posisi node sering berubah-ubah. Hal ini menyebabkan topologi jaringan yang ada di VANET sering berubah.

3. *Unbounded Network size*

VANET bisa di implementasikan untuk satu kota, beberapa kota, bahkan untuk satu negara. Hal ini mengakibatkan ukuran jaringan pada VANET tidak terbatas secara geografis.



4. *Frequent Exchange of Information*

Pada jaringan ad-hoc yang berada di VANET akan memotivasi suatu node yang ada untuk mengumpulkan suatu informasi berupa kendaraan atau keadaan jalan. Karena itu pertukaran informasi terhadap node satu ke node lain menjadi sering.

5. *Wireless Communication*

VANET didesain untuk jaringan *wireless*. Node akan terhubung dan saling bertukar melalui *wireless*. Karena ini keamanan data harus dipertimbangkan untuk berkomunikasi.

6. *Time Critical*

Penyampaian pesan informasi harus dikirimkan ke node tujuan dengan cepat.

7. *Sufficient Energy*

VANET tidak membutuhkan energi yang besar sehingga dapat memberikan energinya ke *On Board Unit*.

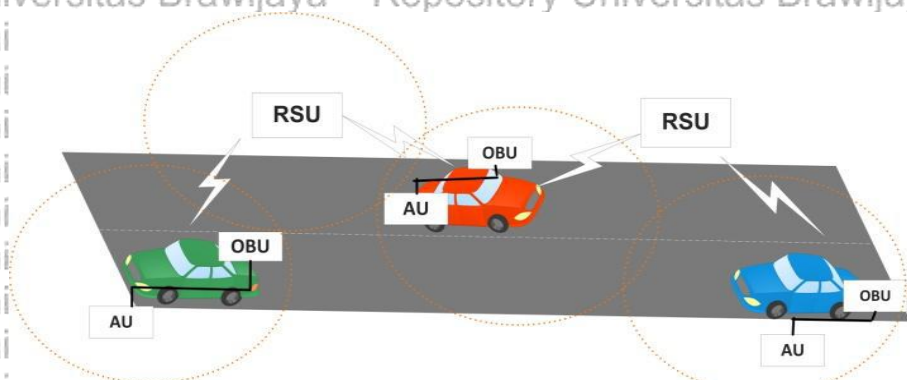
8. *Better Physical Protection*

Node yang ada di VANET mempunyai keamanan yang baik, sehingga sulit diserang atau dapat bertahan dari serangan infrastruktur.

2.3 *Roadside Unit (RSU)*

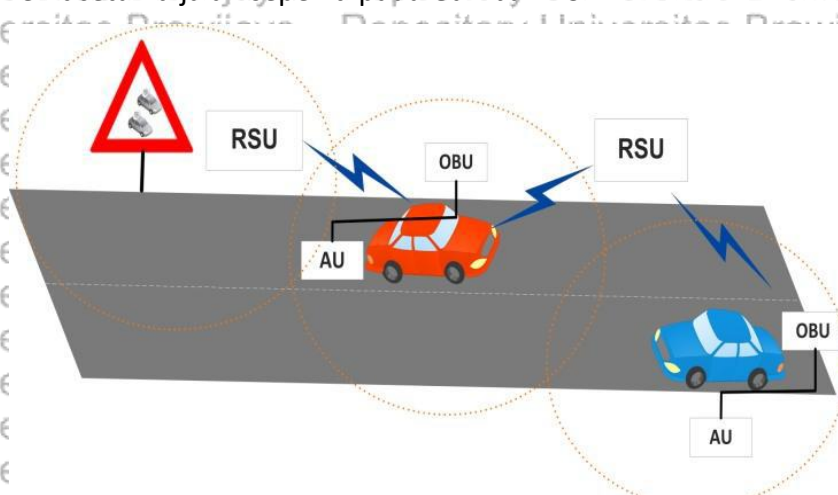
Roadside Unit adalah perangkat yang digunakan sebagai *access point* yang biasanya diletakkan di samping jalan atau lokasi tertentu seperti area parkir. Kegunaan utama RSU adalah seperti ini :

1. Untuk melebarkan jarak komunikasi jaringan ad hoc untuk saling bertukar informasi ke *On Board Unit* (OBU) lain dan dengan bertukar informasi ke RSU lain untuk meneruskannya ke OBU lain seperti pada Gambar 2.2.



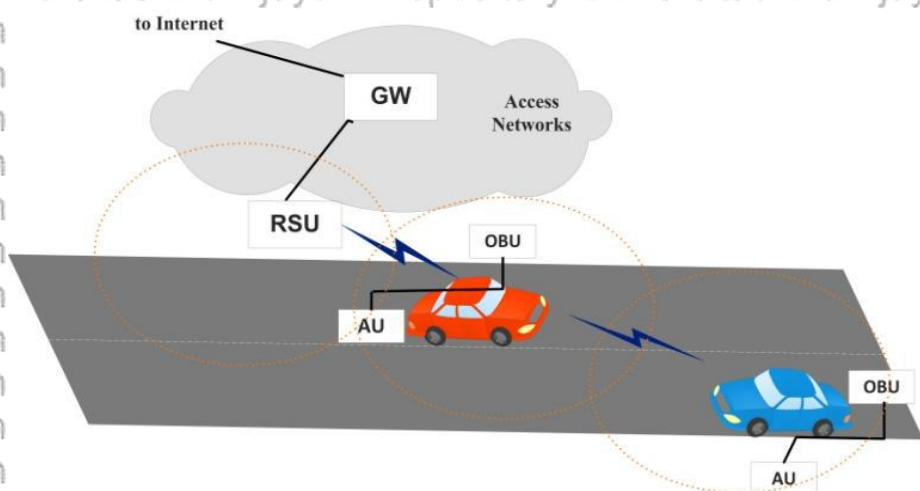
Gambar 2.2 RSU melebarkan jarak jaringan ad hoc dengan meneruskan data OBU

2. RSU harus bisa memberikan informasi tentang keselamatan serta keamanan dalam berkendara seperti adanya lubang, tanjakan, turunan dan kemacetan di jalan seperti pada Gambar 2.3.



Gambar 2.3 RSU memberikan informasi keamanan dan keselamatan dalam berkendara

3. Untuk memberikan koneksi internet ke *On Board Unit* (OBU) seperti pada Gambar 2.4.



Gambar 2.4 RSU memberikan koneksi internet ke OBU

Sumber : (Baskar, & Chawla, 2015)

Manfaat dari penggunaan RSU adalah penyampaian informasi dari kendaraan ke kendaraan bisa lebih cepat sehingga pengemudi dapat menghindari kemacetan sebelum terlambat. Contohnya adalah jika di suatu jalur terdapat kendaraan hendak melewati jalan yang macet, maka kendaraan di sudah terjebak kemacetan akan mengirimkan informasi ke kendaraan tersebut dengan menggunakan bantuan RSU agar informasi lebih cepat sampai sehingga pengemudi akan melewati jalur lain sebelum kendaraan tersebut terjebak kemacetan dan tidak bisa berputar balik. Dalam hal ini RSU berperan sebagai penerus informasi agar informasi itu lebih cepat sampai (Baskar, & Chawla, 2015).



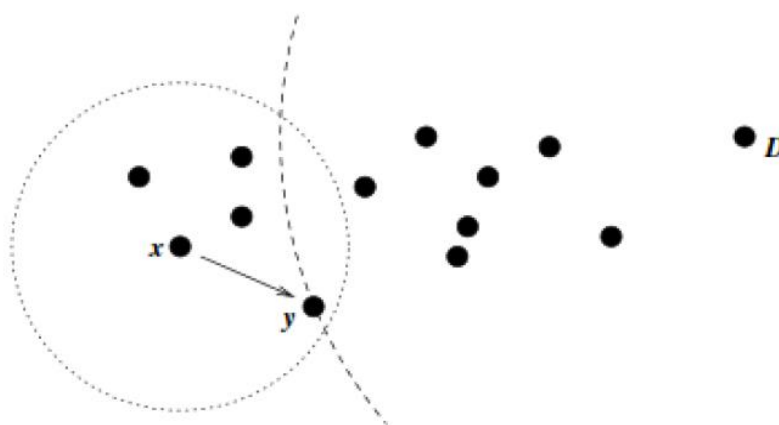
2.4 Greedy Perimeter Stateless Routing (GPSR)

Protokol routing GPSR didasarkan dari posisi router dan mengasumsi setiap node mempunyai akses ke pelayanan lokasi dan mengetahui koordinat posisinya. GPSR membuat keputusan *greedy forwarding* menggunakan hanya dari informasi tentang router tetangga yang ada dalam di topologi jaringan. Hop terbaik akan dianggap node tetangga dengan jarak paling sedikit dari tujuan. Ketika *Greedy forwarding* sekiranya tidak bisa dilanjutkan, maka *Perimeter Forwarding* akan berjalan. Pada routing GPSR, sinyal simpel yang dipancarkan menyediakan informasi semua *node* tetangganya, secara periodik tiap *node* akan mengirimkan suatu sinyal yang membroadcast MAC address yang hanya berisi identitas dan posisi masing-masing. (Bouras, et al., 2015).

Greedy Perimeter Stateless Routing merupakan jenis *geographical routing protocol* yang dalam prosesnya menggunakan *position-based routing*, yang mana tiap node akan saling mengetahui posisi mereka dan tetangga terdekatnya. Protokol routing GPSR akan mengirimkan suatu *hello message* secara periodik untuk mengupdate informasi node yang ada di dalam jangkauan transmisinya. Terdapat 2 metode pada protokol GPSR dalam mengirim paket data, diantaranya

1. Greedy Forwarding

Metode ini adalah metode utama yang akan digunakan untuk pengiriman paket data. *Greedy Forwarding* meneruskan paket data ke node yang dikiranya paling dekat dengan node tujuan. Karena metode ini, protokol GPSR diharapkan dapat dengan cepat mencapai tujuannya karena selalu mencari node terdekat dengan tujuannya.



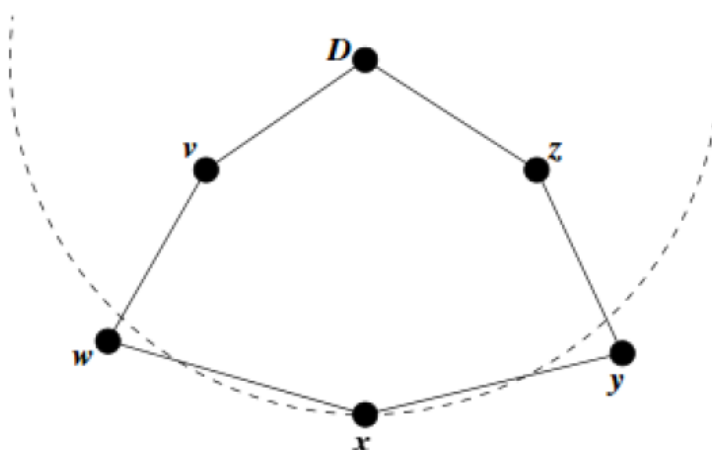
Gambar 2.5 Metode *greedy forwarding*

Sumber : (Karp, & Kung, 2000)



2. Perimeter Forwarding

Metode ini akan bekerja jika metode *greedy forwarding* tidak dapat menemukan node yang berada paling dekat dengan node tujuan. Metode ini memanfaatkan node yang ada di sekitarnya yang sekiranya dapat mencapai ke node tujuan.



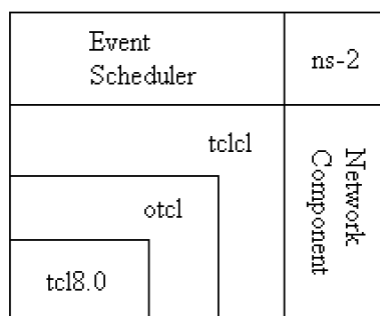
Gambar 2.6 Metode *perimeter forwarding*

(Sumber gambar : Karp, & Kung, 2000)

Berdasarkan Gambar 2.6, source node atau node x tidak mempunyai node tetangga yang berdekatan dengan node tujuan. Maka node x akan memilih node W sesuai aturan jarum jam yang kemudian akan diteruskan ke node v sebagai pengiriman paket sampai akhirnya sampai ke node tujuan. Metode ini memiliki kelambatan karena metode ini memilih jalur lebih panjang dikarenakan melewati banyak hop (Saputra, et al., 2018). Dalam 2 metode yang telah dijelaskan, metode *Greedy Forwarding* lebih baik dari *Greedy perimeter* dikarenakan delay pada *Greedy Perimeter* lebih lama dibanding *Greedy Forwarding*.

2.5 Network Simulator (NS2)

NS 2 adalah simulator jaringan yang bersifat *open source* dan dibuat dengan bahasa C++ yang biasanya digunakan pada *Operating system* (OS) linux.



Gambar 2.7 Arsitektur NS 2



Gambar 2.7 merupakan gambar dari arsitektur simulator NS 2 dimana menggunakan *event scheduler* dan *OTCL script*. Fitur-fitur yang ada di NS 2 adalah sebagai berikut :

1. NS 2 merupakan *software* yang *open source* sehingga semua orang bisa mengunduhnya tanpa membayar.
2. NS 2 adalah simulator yang valid karena mensupport banyak protokol seperti TCP, UDP, FTP, CBR, dll. Sehingga membuat simulator ini sering digunakan untuk meneliti.
3. NS 2 merupakan *user friendly* karena GUI nya.
4. Terdapat 2 representasi grafis yang digunakan di NS2 yang bernama *Network Animator* (Nam) dan Xgraph.
5. Mensupport simulasi *real-time* dan dapat digunakan sebagai bahan pembelajaran (Saluja, et al., 2017).

2.6 Simulation of Urban Mobility (SUMO)

Simulation of Urban Mobility (SUMO) merupakan simulator *open source* yang digunakan untuk mensimulasikan lalu lintas. Selain mensimulasikan lalu lintas SUMO digunakan sebagai simulasi skenario jumlah kendaraan, kepadatan kendaraan dan kecepatan kendaraan (Deshmukh, & Dorle, 2016). SUMO membutuhkan *tools randomTrips.py*, *polyconvert*, dan *netconvert* dalam mengenerate *trafficnya*. *RandomTrips.py* berguna menghasilkan rute yang acak. *Polyconvert* berguna mengimpor bentuk geometris seperti bangunan, kondisi jalanan dll yang digunakan untuk divisualisasikan pada SUMO-GUI. *Netconvert* berguna mengimpor jaringan yang ada di jalanan supaya bisa diidentifikasi node, sinyal, dan junctions sehingga bisa disesuaikan dan kompatibel dengan SUMO.

2.7 Parameter Pengujian

Parameter pengujian yang dipilih dalam menganalisis performa routing GPSR pada VANET dengan menggunakan RSU adalah *throughput*, *end to end delay* dan *packet delivery ratio* (PDR).

2.7.1 Throughput

Throughput adalah kemampuan dari suatu jaringan dalam mengirim data atau kecepatan rata-rata yang dapat diterima pada selang waktu tertentu. Rumus menghitung throughput :

$$\text{Throughput} = \frac{\text{Jumlah data yang dikirim}}{\text{waktu pengiriman data}} \quad (2.1)$$

2.7.2 End to End Delay

End to end delay merupakan waktu yang diperlukan untuk mentransmisikan paket data dari node sumber sampai ke node tujuan. Cara menghitung end to end delay atau rata-rata delay :



$$\text{Delay} = \frac{\text{Paket yang dikirim} - \text{paket yang diterima}}{\text{Total paket yang diterima}}$$

2.7.3 Packet delivery ratio (PDR)

Packet delivery ratio adalah paket data yang diterima oleh tujuan terhadap paket yang dikirimkan dari sumber, atau $\text{PDR} = (\text{paket yang diterima} / \text{paket yang dikirim}) \times 100\%$.

$$\text{PDR} = \frac{\text{Paket yang diterima}}{\text{Paket yang dikirim}} \times 100\% \quad (2.3)$$



BAB 3 METODOLOGI

Pada bab ini menjelaskan metode penelitian yang akan dipakai. Berdasarkan masalah yang dijelaskan pada bab sebelumnya, maka penelitian akan dilakukan adalah menempatkan RSU di peta kota Malang pada simulasi VANET dengan menggunakan protokol GPSR. Pada penelitian simulasi yang akan dilakukan, akan ditempatkan RSU berdasarkan jarak yang ditentukan dengan banyaknya kendaraan dan juga kecepatan kendaraan yang sudah ditentukan. Hasil dari simulasi nantinya akan diuji oleh parameter uji yang datanya bisa dijadikan sebagai perbandingan dalam pemakaian RSU berdasarkan jarak yang akan diuji.

3.1 Studi Literatur

Pada tahap ini akan dijelaskan tentang dasar-dasar teori yang digunakan di penelitian skripsi ini, yang meliputi :

1. *Vehicular Ad Hoc Network* (VANET)

Pada studi literatur VANET mempelajari bagaimana sifat *node* kendaraan pada VANET dan interaksi antar *node* kendaraan.

2. *Roadside Unit* (RSU)

Pada studi literatur RSU mempelajari bagaimana sifat *node* infrastruktur berinteraksi terhadap *node* kendaraan.

3. *Greedy Perimeter Stateless Routing* (GPSR)

Pada studi literatur GPSR mempelajari karakteristik dari protokol tersebut dan bagaimana protokol tersebut menjalankan routing menggunakan *greedy forwarding* dan *perimeter forwarding*.

4. *Network Simulator 2* (NS2)

Pada studi literatur NS2 mempelajari tentang penggunaan tools, objek, dan infrastruktur yang disediakan agar bisa menjalankan simulasi yang sesuai dengan pengujian.

5. *Simulation of Urban Mobility* (SUMO)

Pada studi literatur SUMO mempelajari tentang penggunaan tools agar bisa melihat dan menjalankan pergerakan awal dari skenario kendaraan dengan peta yang diambil.

6. Parameter pengujian

Pada studi literatur parameter pengujian mempelajari teori serta perhitungan agar bisa mendapatkan hasil dari parameter uji yang diantaranya *throughput*, *end-to-end delay*, dan *packet delivery ratio*.

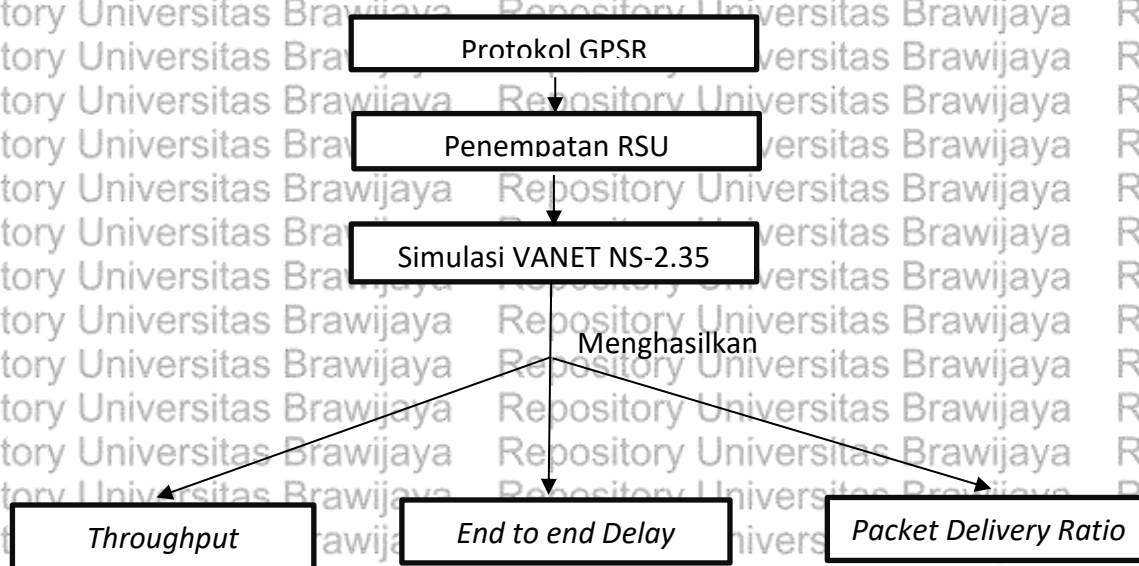


3.2 Perancangan Sistem

Pada tahap ini dijelaskan mengenai gambaran besar tentang topologi beserta skenario yang nantinya akan digunakan pada penelitian yang dilakukan berdasarkan masalah yang sudah dibahas.

3.2.1 Gambaran Umum Sistem

Pada penelitian ini bertujuan untuk melakukan implementasi protokol GPSR pada *Road Side Unit* (RSU) di *Vehicle ad-hoc Network* (VANET) menggunakan simulator NS-2.35. Berikut rancangan simulasi dapat dilihat pada Gambar 4.1



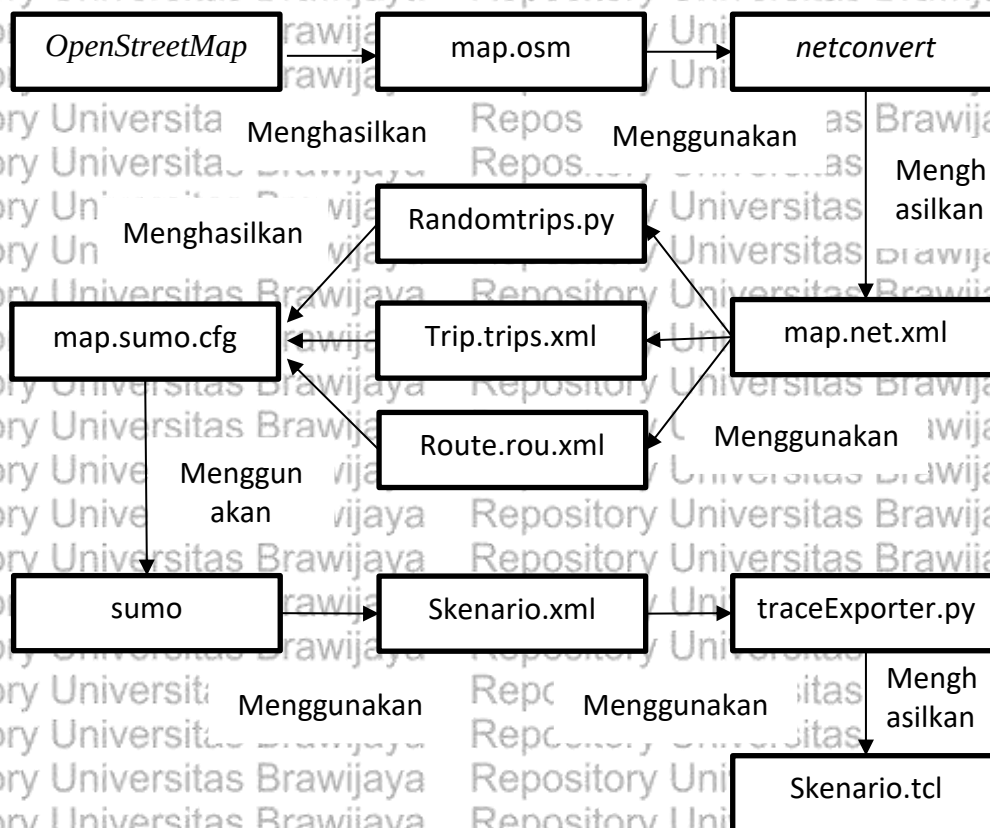
Gambar 3.1 Diagram Rancangan Simulasi

Gambar 3.1 menjelaskan tentang rancangan penelitian yang akan diteliti. Tahap awal dari perancangan ini adalah, merancang jaringan protokol yang akan digunakan yaitu protokol GPSR. Setelah tahap perancangan protokol selesai, akan dilanjutkan penempatan RSU. Setelah perancangan selesai merancang protokol GPSR serta RSU maka akan dilakukan simulasi pada VANET untuk mendapatkan nilai parameter yang diujikan, yaitu *thoroughput*, *end-to-end delay*, dan *packet delivery ratio*. Penilaian pada peneletakan RSU dilakukan dengan menggunakan skema penyebaran *information dissemination objective* yang bertujuan untuk memperbaiki performa seperti mengurangi *end to end delay*, serta menambah *packet throughput*. *Throughput* mengacu pada besar suatu paket data yang dibawa oleh trafik pada suatu jaringan. Besar dari *throughput* akan menunjukkan kualitas dari protokol routing yang digunakan dimana semakin besar nilainya akan semakin baik. *End to end delay* adalah merupakan suatu parameter yang dibutuhkan jika menggunakan suatu routing protokol karena semakin besar nilainya maka kinerja dari routing protokol tersebut akan semakin lambat dimana kualitas dari suatu jaringan tersebut akan menurun. *Packet delivery ratio* merupakan parameter yang dibutuhkan untuk menunjukkan seberapa baik paket

yang diterima dari *node* sumber ke *node* tujuan. Jika nilai *packet delivery ratio* makin tinggi maka kualitas jaringannya semakin baik.

3.2.2 Perancangan Skenario

Perancangan skenario yang digunakan pada penelitian ini adalah simulasi di wilayah Kota Malang dengan menggunakan simulator SUMO. Pengambilan peta wilayah diambil menggunakan *OpenStreetMap*. Keluaran dari hasil *OpenStreetMap* adalah file berupa *.osm* yang formatnya dapat dirubah agar bisa digunakan oleh simulator SUMO dan NS-2. Berikut adalah mekanisme pembuatan peta skenario :



Gambar 3.2 Diagram Alur Perancangan Peta Skenario

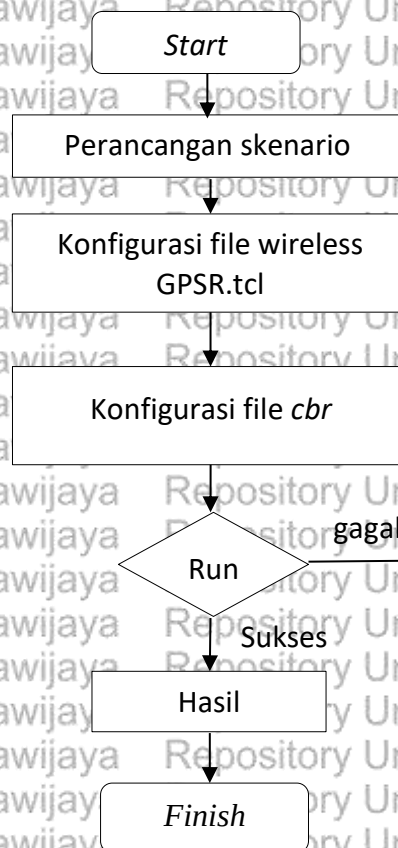
Gambar 3.2 menjelaskan tentang perancangan pembuatan peta skenario kota Malang yang akan dibuat. Tahap pertama adalah memilih peta mana yang mau diambil melalui *Openstreetmap* lalu mengexport peta tersebut menjadi format *.osm*. Tahap berikutnya adalah mengkonvert file *.osm* menjadi *.net.xml* menggunakan tool *netconvert* agar bisa diproses untuk bisa digunakan pada SUMO serta dalam tahap ini *default speed* pada *node* yang akan dibuat akan ditentukan. Tahap selanjutnya yang dilakukan adalah pembuatan *node* menggunakan tools *randomTrips.py* yang didalamnya terdapat tools *trip.trips.xml* serta pembuatan rute dengan menggunakan tools *route.rou.xml*. Hasil dari *map.net.xml* menggunakan 3 tools tersebut adalah *map.sumo.cfg* yang barulah bisa dijalankan di simulator SUMO. Jika SUMO sudah berhasil



menjalankan *map.sumo.cfg* yang dibuat, maka file tersebut akan dikonvert ke dalam bentuk *skenario.xml* dahulu. Dengan menggunakan tools *traceExporter.py* file tersebut akan diubah menjadi *skenario.tcl* yang bisa digunakan dalam NS-2.

3.2.3 Perancangan *Greedy Perimeter Stateless Routing (GPSR)*

Dalam perancangan protokol GPSR ini akan dijelaskan tahapan perancangan protokol dengan menggunakan *node*, serta kecepatan *node* yang sudah didapatkan. Berikut flowchart perancangan protokol GPSR.



Gambar 3.3 Flowchart Protokol GPSR

Gambar 3.3 menjelaskan tentang *flowchart* protokol *Greedy Perimeter Stateless Routing (GPSR)*. Hal pertama yang dilakukan adalah menentukan peta kota Malang yang akan dipakai, lalu membuat *node* kendaraan yang berjumlah 140, 160, dan 180, dan kecepatan maksimal *node* yang dipakai adalah 40km/jam, 50km/jam, dan 60km/jam. Selanjutnya adalah konfigurasi di file wireless *GPSR.tcl* untuk menseting wireless GPSR termasuk waktu simulasi yang digunakan, jumlah data, serta *transmission range* yang akan digunakan. Selanjutnya adalah Konfigurasi file *cbr* bertujuan untuk menentukan *node* mana saja yang akan diujikan dan diambil nilainya. Untuk tahapan selanjutnya adalah menjalankan simulator, jika simulasinya sukses maka akan mendapatkan hasil berupa *throughput*, *end to end delay*, dan *packet delivery ratio* yang nantinya akan dianalisis. Cara kerja *routing* protokol GPSR adalah dia akan mengirim pesan *hello* ke seluruh *node* tetangganya secara periodik dengan mengirimkan posisi dan ID

mereka sehingga tiap *node* mengetahui posisi mereka pada topologi jaringan tersebut. Yang nantinya proses pengiriman data akan terbentuk hubungan antara kendaraan dengan kendaraan (V2V) dan kendaraan dengan infrastruktur (V2I) atau (I2V).

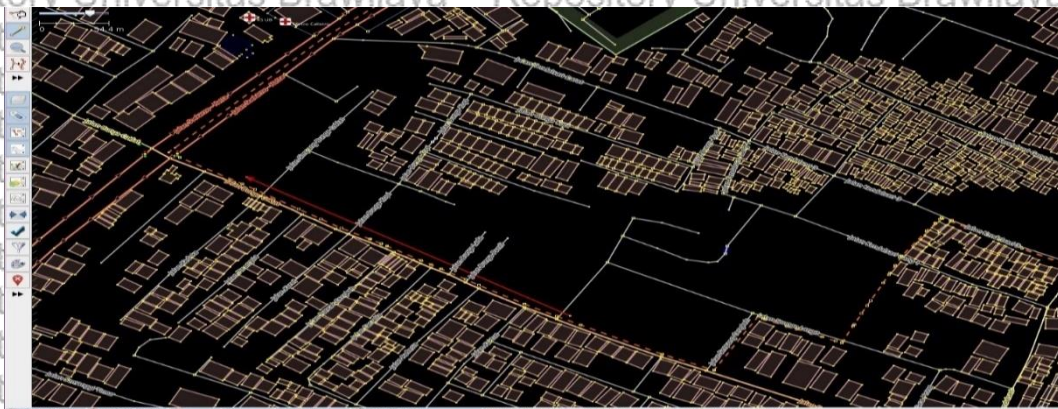
3.2.4 Perancangan Peletakan *Roadside Unit* (RSU)

Pada sub bab perancangan ini akan menjelaskan perancangan *node* yang akan digunakan yaitu RSU. Penempatan RSU akan ditempatkan berdasarkan jarak antar RSU adalah 300m, 400m, dan 500m. Penggunaan RSU berjarak 300, 400, 500m adalah karena berdasarkan penelitian yang dilakukan Bastian et al pada tahun 2017 menggunakan *transmission range* 250m dengan ukuran map 1500x1500 mendapatkan hasil yang cukup baik dimana ukuran mapnya tidak jauh berbeda dari penelitian ini yaitu 2200x 1300m, penambahan jarak 50m agar RSU tidak berkomunikasi ke sesama RSU supaya mendapatkan data yang lebih valid. Jika jarak tiap RSU terlalu jauh dari *transmission range* maka sulit untuk mendapatkan data yang valid. pada penelitian RSU berfungsi sebagai sumber informasi sehingga kendaraan bisa mengenal suatu keadaan di jalan dengan cepat.



Gambar 3.4 Gambar Openstreet Map

Gambar 3.4 adalah map area kota Malang bagian soekarno hatta dan sekitarnya yang diambil menggunakan *openstreet map* yang akan digunakan sebagai map dasar untuk simulasi. Map yang sudah diambil lalu diubah ekstensinya menjadi .osm akan dibuka melalui *Java Openstreet Map* (JOSM) untuk mengukur jarak RSU yang akan digunakan.



Gambar 3.5 Gambar Pengambilan Letak Jarak RSU



Gambar 3.5 menjelaskan bagaimana pengambilan letak jarak tiap RSU yang akan digunakan. Pengambilan jarak diukur dari RSU ke RSU lain yang ditandai dengan tanda panah berwarna merah. RSU biasanya diletakkan di dekat persimpangan jalan, akan tetapi jika diletakkan di tiap persimpangan jalan maka akan menjadi kurang efektif dilihat dari segi fungsi serta biaya. Untuk itu diperlukan penelitian untuk menentukan jarak yang optimal dan situasi jalan seperti apa yang baik digunakan. Pada penelitian ini RSU diletakkan di jalur yang sering dilewati kendaraan karena dalam keadaan mobilitas yang selalu berubah, *node* kendaraan akan terbantu jika terdapat RSU sebagai *node* yang diam meneruskan informasi dan mengembalikan informasi tersebut kembali ke *node* sumber.



Gambar 3.6 Gambar Topologi Peneletakan RSU Tiap 300m

Gambar 3.6 menjelaskan gambaran dari peneletakan RSU yang ditandai dengan dot-dot yang ditaruh di persimpangan jalan. Dalam peneletakannya, tiap RSU diletakkan berjarak 300m sepanjang jalan raya.



Gambar 3.7 Gambar Topologi Peneletakan RSU Tiap 400m

Gambar 3.7 menjelaskan gambaran dari peneletakan RSU yang ditandai dengan dot-dot yang ditaruh di persimpangan jalan. Dalam peneletakannya, tiap RSU diletakkan berjarak 400m sepanjang jalan raya.



Gambar 3.8 Gambar Topologi Peneletakan RSU Tiap 500m

Gambar 3.8 menjelaskan gambaran dari peneletakan RSU yang ditandai dengan dot-dot yang ditaruh di persimpangan jalan. Dalam peneletakannya, tiap RSU diletakkan berjarak 500m sepanjang jalan raya.

3.2.5 Perancangan Simulasi di *Network Simulator (NS-2)*

Pada perancangan yang dilakukan yaitu simulasi VANET di *Network Simulator*, dilakukan dengan menggunakan skenario yang sudah dibuat. Berikut merupakan Tabel parameter simulasi yang akan diujikan.

No	Parameter	Spesifikasi
1	Simulator	<i>Network Simulator (NS-2)</i>
2	Protokol	<i>GPSR</i>
3	Area simulasi	2200 x 1300m
4	Waktu simulasi	1000 detik
5	Jumlah <i>node</i> kendaraan	140, 160 dan 180
6	Tipe data	<i>Constant Bit Rate (CBR)</i>
7	Jumlah RSU	11, 10, dan 8
8	Peta	Peta nyata
9	<i>Packet Size</i>	512 Byte
10	<i>Transmission Range</i>	250m
11	Kecepatan kendaraan	40km/jam, 50km/jam, dan 60km/jam

Tabel 3.1 Parameter Simulasi



Pada Tabel 3.1 ditetapkan simulasi ini menggunakan waktu simulasi 1000 detik agar mendapat data yang lebih valid. Tipe data yang digunakan adalah *constant bit rate* (CBR) yang termasuk jenis UDP agar informasi lebih cepat tersampaikan. Jumlah *node* kendaraan adalah 140, 160, dan 180 dengan asumsi pada peta yang digunakan sedang tidak terlalu padat dengan kendaraan. Jumlah RSU yang digunakan adalah 11 untuk RSU tiap 300m, 10 untuk RSU tiap 400m dan 8 untuk RSU tiap 500m.

Setelah perancangan skenario, protokol, serta penempatan RSU sudah dilakukan sesuai tabel parameter simulasi, maka simulasi dilakukan untuk mendapatkan hasil *throughput*, *end-to-end delay*, dan *packet delivery ratio* yang nantinya akan dianalisis. Untuk bisa mendapatkan hasil tersebut, diperlukan file berke ekstensi *.awk* yang dapat digunakan di ns-2. Berikut alur perancangannya :



Gambar 3.8 Alur perancangan file .awk

Gambar 3.8 menjelaskan bahwa untuk merancang dan menggunakan file berke ekstensi *.awk* diperlukan *trace file* yang berisi hasil data simulasi.

3.3 Implementasi

Pada tahap ini, proses implementasi dilakukan setelah topologi selesai dibuat. Lalu pada topologi jaringan yang telah dibuat akan dilakukan implementasi agar pengujian dapat dilakukan yang meliputi :

1. Konfigurasi perancangan protokol, jumlah *node*, mobilitas *node*, *node* RSU, dan *wireless* pada jaringan VANET.
2. Menjalankan simulasi sesuai dengan skenario yang diujikan yaitu berdasarkan jumlah *node* kendaraan, dan kecepatan kendaraan.

3.4 Pengujian dan Analisis

Pada tahap ini dilakukan pengujian sistem yang sudah dibuat, dan analisis dari hasil simulasi yang berasal dari simulator NS 2 dengan parameter ujinya adalah *throughput*, *end to end delay*, *packet delivery ratio* (PDR) pada VANET dengan menggunakan protokol GPSR yang terdapat RSU. Berikut skenario yang akan diuji pada penelitian ini :



Skenario	Jumlah Kendaraan	Kecepatan Kendaraan	Jarak antar RSU	Ukuran Map	Waktu Pengujian
Kepadatan Kendaraan	140, 160, 180	40 km/jam	300, 400, 500 m	2200 x 1300 m	1000 detik
Kecepatan Kendaraan	140	40, 50, 60 km/jam	300, 400, 500 m	2200 x 1300 m	1000 detik

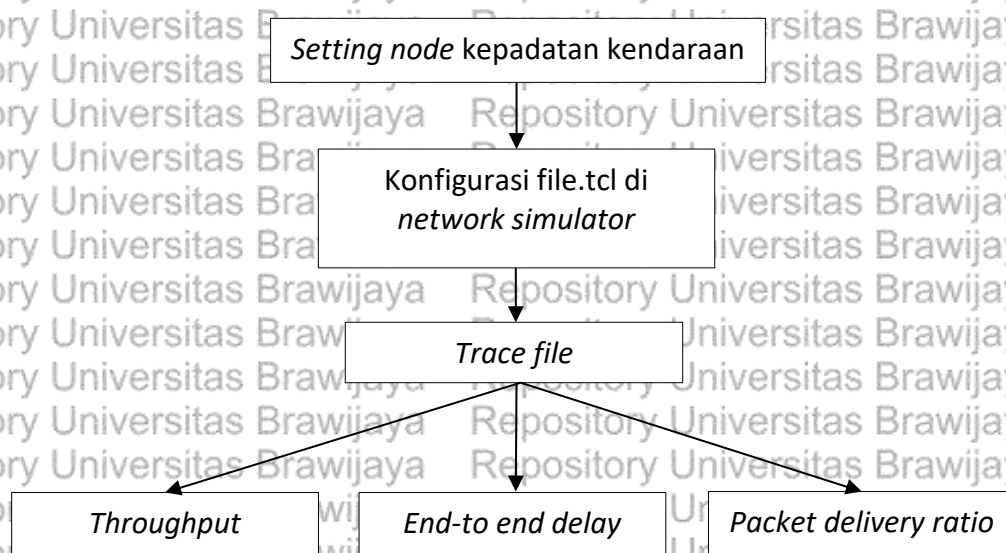
Tabel 3.2 Skenario Pengujian

Pada Tabel 3.2 terdapat dua skenario yang akan diuji yaitu skenario berdasarkan banyaknya kendaraan dan kecepatan kendaraan. Pada skenario kepadatan kendaraan akan didapatkan hasil apakah jumlah kepadatan kendaraan serta dengan memakai RSU dengan jarak yang berbeda mempengaruhi hasil dari nilai parameter dengan kecepatan kendaraan yang sama yaitu 40 km/jam.

Pada skenario kecepatan kendaraan akan diuji dan didapatkan hasil apakah kecepatan kendaraan dengan memakai RSU dengan jarak yang berbeda mempengaruhi hasil dari nilai parameter dengan jumlah kendaraan yang sama yaitu 140.

3.4.1 Pengujian Terhadap Kepadatan Kendaraan

Pada Tabel 3.2 pengujian yang akan dilakukan adalah berdasarkan kepadatan kendaraan terhadap parameter yang diuji dengan jumlah 140, 160, dan 180 yang mempunyai kecepatan 40 km/jam dengan memakai RSU yang jaraknya tiap 300m, 400m, dan 500m pada ukuran map 2200x1300 m dalam waktu 1000 detik.



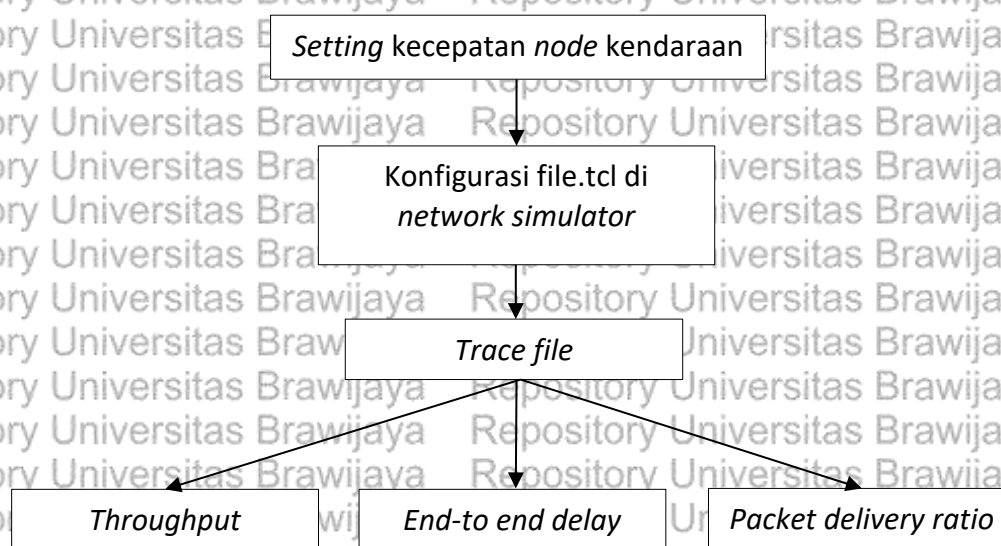
Gambar 3.9 Flowchart Perancangan Pengujian Berdasarkan Banyak Kendaraan



Gambar 3.9 menjelaskan flowchart bagaimana merancang pengujian berdasarkan banyaknya kendaraan. Pertama adalah mensetting banyaknya jumlah pada suatu *node* kendaraan yang sesuai dengan parameter simulasi yang sudah ditentukan. Setelah itu mengkonfigurasi file.tcl sesuai dengan langkah sebelumnya dan parameter simulasi yang telah ditentukan. Setelah menjalankan simulasi, hasilnya akan berupa *trace file* yang digunakan untuk menentukan *Throughput*, *end-to-end delay*, dan *packet delivery ratio*.

3.4.2 Pengujian Terhadap Kecepatan Kendaraan

Pada Tabel 3.2 pengujian dilakukan berdasarkan kecepatan kendaraan terhadap paramete yang diuji dengan kecepatan 40km/jam, 50km/jam, dan 60km/jam yang mempunyai jumlah kendaraan 140 dengan memakai RSU yang jaraknya tiap 300m, 400m, dan 500m pada ukuran map 2200x1300 m dalam waktu 1000 detik.



Gambar 3.10 Flowchart Perancangan Pengujian Berdasarkan Kecepatan Kendaraan

Gambar 3.10 menjelaskan flowchart bagaimana merancang pengujian berdasarkan kecepatan kendaraan. Pertama adalah mensetting kecepatan *node* kendaraan sesuai dengan parameter simulasi yang sudah ditentukan. Setelah itu mengkonfigurasi file.tcl sesuai dengan langkah sebelumnya dan parameter simulasi yang telah ditentukan. Setelah menjalankan simulasi, hasilnya akan berupa *trace file* yang digunakan untuk menentukan *Throughput*, *end-to-end delay*, dan *packet delivery ratio*.

3.5 Kesimpulan dan Saran

Pada tahap ini dilakukanlah penarikan berupa suatu kesimpulan berdasarkan rumusan masalah yang didapat berdasarkan dari hasil analisis pengujian yang sudah didapatkan sebelumnya. Saran berguna untuk referensi yang ditujukan kepada pembaca untuk melakukan penelitian lebih lanjut.



BAB 4 IMPLEMENTASI

Pada bab ini proses implementasi sistem akan digunakan dalam proses penelitian. Pada proses implementasi akan dimulai dari analisis kebutuhan dan alur sistem yang mengacu pada bab perancangan.

4.1 Implementasi Skenario

Implementasi skenario dilakukan setelah mendapatkan map dari *Openstreetmap* yang berada di bab perancangan sebelumnya. File yang didapat dari *Openstreetmap* berupa *.osm* dan tidak bisa di run di SUMO. Untuk bisa dijalankan di SUMO, file *.osm* perlu dikonversi kedalam *net.xml*. Untuk bisa mengkonversi menjadi *net.xml*, perlu menggunakan *tools* yang ada di SUMO yang berupa *netconvert*. Dalam pembuatan kecepatan node, dibatasi sampai 40 km/jam saja.

```
netconvert --default.speed=40 --osm-files map.osm -o map.net.xml
```

Tabel 4.1 Perintah mengkonversi file OSM

Langkah berikutnya yang dilakukan adalah pembuatan node. Pada langkah pembuatan node ini di *setting* sesuai dengan skenario jumlah *node* yaitu, 140, 160 dan 180. Penggunaan dari *tools* yang digunakan adalah *tools* dari SUMO yaitu *randomtrips*.

```
python /usr/local/src/sumo-0.32.0/tools/randomTrips.py -n  
map.net.xml --trip-attributes="departLane=\"best\"  
departSpeed=\"max\" departPos=\"random_free\" -r map.rou.xml -e  
140 -1
```

Tabel 4.2 Perintah membuat node dengan tools randomTrips.py

Angka 140 yang tertera diatas adalah jumlah node yang akan di konfigurasi pada SUMO. Untuk *departSpeed* adalah sebuah fungsi untuk menentukan kecepatan kendaraan, karena sudah disetting kecepatan maksimal adalah 40 km/jam maka kecepatan kendaraan tidak akan melebihi 40 km/jam. Hasil keluaran dari kode diatas adalah file *trip.trips.xml*. Jika kode diatas dijalankan, maka *tools duarouter* akan dijalankan juga secara otomatis.

Dalam menjalankan suatu file pada SUMO, diperlukan ekstensi *sumo.cfg* agar bisa dijalankan di simulator. oleh karena itu perlu dilakukan pembuatan *gedit peta.sumo.cfg* dengan memasukkan file berupa *map.net.xml* serta *map.rou.xml*.

```
Gedit map.sumo.cfg
```

Tabel 4.3 Perintah membuat file map.sumo.cfg

```
<?xml version="1.0" encoding="UTF-8"?>  
<configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-  
instance"
```




```

xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/sumoCon
figuration.xsd">
<input>
<net-file value="map.net.xml"/>
<route-files value="map.rou.xml"/>
</input>
<time>
<begin value="0"/>
<end value="1000"/>
</time>
</configuration>

```

Tabel 4.4 konfigurasi file map.sumo.cfg

Tahap berikutnya adalah menjalankan file konfigurasi diatas pada SUMO untuk mengetahui *node* dan rute yang telah dibuat tadi berjalan atau tidak dengan perintah *sumo-gui*.

```
Sumo-gui map.sumo.cfg
```

Tabel 4.5 Perintah menjalankan map.sumo.cfg pada SUMO

Untuk bisa dijalankan pada *Network Simulator(NS)* file harus di konversi menjadi ekstensi .tcl, namun sebelum dikonversi kedalam file tersebut file *peta.sumo.cfg* harus dikonversi terlebih dahulu kedalam ekstensi.xml.

```
Sumo -c map.sumo.cfg --fod-output skenario.xml
```

Tabel 4.6 Perintah mengkonversi map.sumo.cfg ke skenario.xml

Setelah berhasil di konversi, Tahap berikutnya adalah *convert* file .xml menjadi file yang bisa berjalan pada *Network Simulator(NS)* yaitu .tcl menggunakan *tools traceExporter*.

```
python /usr/local/src/sumo-0.32.0/tools/traceExporter.py --fcd-
input map.sumo.xml --ns2config-output map.tcl --ns2activity-
output activity.tcl --ns2mobility-output mobility.tcl
```

Tabel 4.7 Perintah mengkonversi (.xml) ke (.tcl)

4.2 Implementasi Road Side Unit (RSU)

RSU diimplementasikan sebagai *Intersection node* untuk membantu dalam pencarian node. Node RSU diberikan kecepatan 0.00 supaya *node* itu tidak bergerak. Untuk membedakan yang mana *node RSU* saat simulasi, maka node RSU diberikan label dan warna.

```

1 $node_1(138) set X_ 1169.12
2 $node_1(138) set Y_ 502.54
3 $node_1(138) set Z_ 0
4 $node_1(138) label "RSU"
5 $node_1(138) color "blue"
6 $ns_ at 0.0 "$node_1(138) color blue"
7 $ns_ at 0.0 "$node_1(138) setdest 1169.12 502.54 0.00"

```




8	\$node_(139) set X 875.09
9	\$node_(139) set Y 688.31
10	\$node_(139) set Z 0
11	\$node_(139) label "RSU"
12	\$node_(139) color "blue"
13	\$ns_at 0.0 "\$node_(139) color blue"
14	\$ns_at 0.0 "\$node_(139) setdest 875.09 688.31 0.00"
15	\$node_(140) set X 1196.18
16	\$node_(140) set Y 1018.01
17	\$node_(140) set Z 0
18	\$node_(140) label "RSU"
19	\$node_(140) color "blue"
20	\$ns_at 0.0 "\$node_(140) color blue"
21	\$ns_at 0.0 "\$node_(140) setdest 1196.18 1018.01 0.00"
22	\$node_(141) set X 1447.15
23	\$node_(141) set Y 1243.14
24	\$node_(141) set Z 0
25	\$node_(141) label "RSU"
26	\$node_(141) color "blue"
27	\$ns_at 0.0 "\$node_(141) color blue"
28	\$ns_at 0.0 "\$node_(141) setdest 1447.15 1243.14 0.00"

Tabel 4.8 Implementasi Node RSU

4.3 Implementasi Simulasi NS2

Dalam melakukan simulasi di NS2, diperlukan file .tcl yang berisikan tentang konfigurasi untuk tiap node dan sekumpulan *event* yang harus diatur supaya dapat berjalan. File-file gpr yang perlu diatur ada 3 file yaitu, wireless-gpr.tcl untuk mensetting konfigurasi wireless, cbr100.tcl untuk menentukan node mana yang akan melakukan pengiriman data serta penerima datanya dan mobility.tcl adalah *node* kendaraan. Berikut hasil konfigurasi file tcl:

1	set_opt(chan)	Channel/WirelessChannel
2	set_opt(prop)	Propagation/TwoRayGround
3	set_opt(netif)	Phy/WirelessPhy
4	set_opt(mac)	Mac/802_11
5	set_opt(ifq)	Queue/DropTail/PriQueue;
6	set_opt(ll)	LL
7	set_opt(ant)	Antenna/OmniAntenna
8		
9	set_opt(x)	3500 ;# X dimension of the
10	topography	
11	set_opt(y)	3500 ;# Y dimension of the
12	topography	
13	set_opt(cp)	"./cbr100.tcl"
14	set_opt(sc)	"./mobility.tcl"
15		
16	set_opt(ifqlen)	50 ;# max packet in ifq
17	set_opt(nn)	147 ;# number of nodes
18	set_opt(seed)	0.0
19	set_opt(stop)	1000.0 ;#
20	simulationtime	
21	set_opt(tr)	gpr.tr ;# trace file
22	set_opt(nam)	gpr.nam
23	set_opt(rp)	gpr ;# routing
24	protocol script	



25	set opt (lm) "off" ;# log movement
26	
27	Phy/WirelessPhy set Pt_ 0.2818 ;# 250m
28	# configure
29	\$ns_ node-config -adhocRouting gpsr \
30	-llType \$opt (ll) \
31	-macType \$opt (mac) \
32	-ifqType \$optl (ifq) \
33	-ifqlen \$opti (ifqlen) \
34	-antType \$optl (ant) \
35	-propType \$opt (prop) \
36	-phyType \$opt (netif) \
37	-channelType \$opt (chan) \
38	-topoInstance \$stopoi \
39	-agentTrace ON \
40	-routerTrace ON \
41	-macTrace ON \
	-movementTrace ON

Tabel 4.9 Konfigurasi *wireless-gpsr.tcl*

Pada Tabel 5.9 menjelaskan konfigurasi *wireless-gpsr.tcl* yang dipakai *set opt(nn)* merupakan *Node* keseluruhan yang dipakai berjumlah 147 termasuk dan sudah termasuk *RSU*. *Phy/WirelessPhy set Pt_ 0.2818* digunakan untuk menset *transmission range* berjarak 250m. *AgentTrace* dan *routerTrace* diaktifkan guna mencatat aktifitas yang berlangsung pada *agent protokol routing* dan *routing protokol* yang nantinya catatan tersebut akan tercatat di *trace file*.

1	# GPSR routing agent settings
2	GPSR routing agent settings
3	for {set i 0} {\$i < \$opt (nn)} {incr i} {
4	\$ns_ at 0.00002 "\$ragent_(\$i) turnon"
5	\$ns_ at 20.0 "\$ragent_(\$i) neighborlist"
6	\$ns_ at 30.0 "\$ragent_(\$i) turnoff"
7	}
8	set udp_ (0) [new Agent/UDP]
9	\$ns_ attach-agent \$node_ (30) \$udp_ (0)
10	
11	set null_ (0) [new Agent/Null]
12	\$ns_ attach-agent \$node_ (9) \$null_ (0)
13	
14	set cbr_ (0) [new Application/Traffic/CBR]
15	\$cbr_ (0) set packetSize_ 512
16	\$cbr_ (0) set interval_ 2.0
17	\$cbr_ (0) set random_ 1
18	# \$cbr_ (2) set maxpkts_ 2
19	\$cbr_ (0) attach-agent \$udp_ (0)
20	\$ns_ connect \$udp_ (0) \$null_ (0)
21	\$ns_ at 1.0 "\$cbr_ (0) start"
22	\$ns_ at \$opt (stop) "\$cbr_ (0) stop"

Tabel 4.10 Konfigurasi *cbr100.tcl*

Pada Tabel 5.10 *node* yang menjadi sumber informasi adalah *node 30* dan yang menjadi *node* tujuan adalah *node 9*. Tipe data yang digunakan adalah CBR yang sifatnya mendukung jalannya transmisi sepanjang komunikasi, tetapi CBR membutuhkan toleransi delay karena itu agent yang digunakan adalah UDP mempunyai karakteristik salah satunya adalah *connection less* yang tidak



memerlukan proses negoisasi dalam pengiriman pesan. *Packetsize* yang digunakan adalah 512 byte karena paket yang berukuran kecil kemungkinan akan di drop supaya lebih cepat dalam mengirimkan data.

1	\$node_(20) set X_ 1536.13
2	\$node_(20) set Y_ 868.55
3	\$node_(20) set Z_ 0
4	\$node_(21) set X_ 851.64
5	\$node_(21) set Y_ 1185.74
6	\$node_(21) set Z_ 0
7	\$node_(22) set X_ 1845.92
8	\$node_(22) set Y_ 1526.61
9	\$node_(22) set Z_ 0
10	\$node_(23) set X_ 687.2
11	\$node_(23) set Y_ 1643.97
12	\$node_(23) set Z_ 0
13	\$node_(24) set X_ 878.06
14	\$node_(24) set Y_ 879.78
15	\$node_(24) set Z_ 0
16	\$node_(25) set X_ 2326.67
17	\$node_(25) set Y_ 674.49
18	\$node_(25) set Z_ 0

Tabel 4.11 Konfigurasi *mobility.tcl*

Tabel 4.11 merupakan potongan kode dari *mobility.tcl* yang menunjukkan saat munculnya *node* dengan koordinat yang ditandai dengan *set X* dan *set Y* dan kecepatan dengan *set Z*.

4.4 Implementasi Pengujian

Hasil yang didapat dari simulasi yang telah dilakukan pada NS2 adalah 2 file yang merupakan *trace file* yang berbeda yaitu file yang berekstensi *.nam* dan *.tr*. Hasil dari ekstensi *.nam* akan memperlihatkan simulasi pergerakan *node* yang sudah diatur dan dijalankan melalui *simulator* NAM. Sedangkan ekstensi *.tr* menghasilkan file proses *event* simulasi yang terjadi. Dibawah ini merupakan contoh dari *trace file* yang didapat.

```
Ns wireless-gpsr.tcl
```

Tabel 4.12 Perintah menjalankan *Script .tcl* pada *wireless-gpsr*

1	s 1.0000000000 1 AGT --- 150 cbr 512 [0 0 0 0] -----
2	[1:0 5:0 32 0] [0] 0 0
3	r 1.0000000000 1 RTR --- 150 cbr 512 [0 0 0 0] -----
4	[1:0 5:0 32 0] [0] 0 0
5	s 1.0000000000 1 RTR --- 150 gpsr 69 [0 0 0 0] -----
6	[1:255 5:255 32 0]
7	M 1.00000 0 (1246.86, 1179.06, 0.00), (1240.03, 1173.02),
8	9.12
9	M 1.00000 1 (978.42, 777.08, 0.00), (978.42, 777.08),
10	13.89
11	s 1.000355000 1 MAC --- 150 gpsr 127 [0 ffffffff 1
12	800] --- [1:255 5:255 32 0]
13	r 1.001371283 5 MAC --- 150 gpsr 69 [0 ffffffff 1 800]
14	--- [1:255 5:255 32 0]
15	r 1.001371304 89 MAC --- 150 gpsr 69 [0 ffffffff 1
16	800] --- [1:255 5:255 32 0]



17	r	1.001371372	124	MAC	---	150	gpsr	69	[0	ffffff	1
18		800]									
19	r	1.001371454	138	MAC	---	150	gpsr	69	[0	ffffff	1
20		800]									
21	r	1.001371510	134	MAC	---	150	gpsr	69	[0	ffffff	1
22		800]									
23	r	1.001371640	39	MAC	---	150	gpsr	69	[0	ffffff	1
24		800]									
25	r	1.001371759	99	MAC	---	150	gpsr	69	[0	ffffff	1
26		800]									
27	r	1.001396283	5	RTR	---	150	gpsr	69	[0	ffffff	1
28		800]									
29	r	1.001396304	89	RTR	---	150	gpsr	69	[0	ffffff	1
30		800]									
31	r	1.001396304	89	RTR	---	150	gpsr	69	[0	ffffff	1
32		800]									
33	r	1.001396372	124	RTR	---	150	gpsr	69	[0	ffffff	1
34		800]									
35	r	1.001396454	138	RTR	---	150	gpsr	69	[0	ffffff	1
36		800]									

Tabel 4.13 Hasil Trace File

Berdasarkan Tabel 4.13 makna dari baris pertama yaitu s 2.488538641_99_RTR --- 293 gpsr 29 [0 0 0 0] ----- [99:255 1:255 32 0] adalah pada saat detik 2.488538641 *node* 99 sebagai pengirim data melakukan routing dengan protokol gpsr dan paket sebesar 29 byte. Pada baris 19 yaitu r 2.489474927_8_RTR --- 293 gpsr 29 [0 fffffff 63 800] ----- [99:255 -1:255 32 0] mempunyai makna pada saat detik 2.489474927 *node* 8 mendapatkan paket dengan ip sumber adalah 99 dan ip tujuan adalah 1 dengan masing masing portnya adalah 255.

Setelah menjalankan script .tcl pada wireless-gpsr maka akan mendapatkan hasil *trace* file dan parameter uji pun dapat dianalisa. Terdapat 3 parameter yang akan diujikan di penelitian ini yaitu, *throughput*, *end-to-end delay* dan *packet delivery ratio*.

4.4.1 Implementasi Throughput

Throughput merupakan kemampuan dari suatu jaringan dalam mengirim data atau kecepatan rata-rata yang dapat diterima pada selang waktu tertentu. Nilai *throughput* dapat didapatkan dengan menjalankan perintah berikut yang script *awk* nya ada di lampiran.

```
Awk -f throughput.awk gpsr.tr
```

Tabel 4.14 Perintah menjalankan Script .awk pada throughput

4.4.2 Implementasi End-to-end Delay

End-to-end delay merupakan waktu yang diperlukan oleh paket data dari node sumber sampai ke node tujuan. Nilai *end-to-end* dapat didapatkan dengan menjalankan perintah berikut yang script *awk* nya ada di lampiran.

```
Awk -f end-to-end delay.awk gpsr.tr
```

Tabel 4.15 Perintah menjalankan Script .awk pada End-to-end delay



4.4.3 Implementasi *Packet Delivery Ratio*

Packet delivery ratio adalah paket data yang diterima oleh tujuan terhadap paket yang dikirimkan dari sumber. Nilai pada suatu *packet delivery ratio* dapat didapatkan dengan menjalankan perintah berikut yang *script .awk* nya ada di lampiran.

```
Awk -f pdr.awk gpsr.tr
```

Tabel 4.16 Perintah menjalankan *Script .awk* pada *Packet Delivery Ratio*



BAB 5 PENGUJIAN DAN ANALISIS

Bab ini membahas hasil pengujian yang telah disimulasikan yaitu pengujian terhadap kepadatan kendaraan dan pengujian terhadap kecepatan kendaraan dengan parameter uji berupa *throughput*, *end-to-end delay* dan *packet delivery ratio*. Berikut adalah contoh dari simulasi *nam* yang dijalankan.



Gambar 5.1 Peletakkan RSU di Simulator *Nam*

Gambar 5.1 menunjukkan lokasi peletakkan RSU di simulator *nam* sebelum dimunculkannya *node* kendaraan.

5.1 Pengujian Terhadap Kepadatan Kendaraan

Di sub bab ini akan dijelaskan hasil dari simulasi pengujian yang dilakukan di area kota Malang. Hasil dari pengujian akan didapatkan berdasarkan kepadatan kendaraan yang mempunyai jumlah *node* 140, 160, dan 180 dengan RSU yang topologinya mengacu ke Gambar 3.6, Gambar 3.7, dan Gambar 3.8.

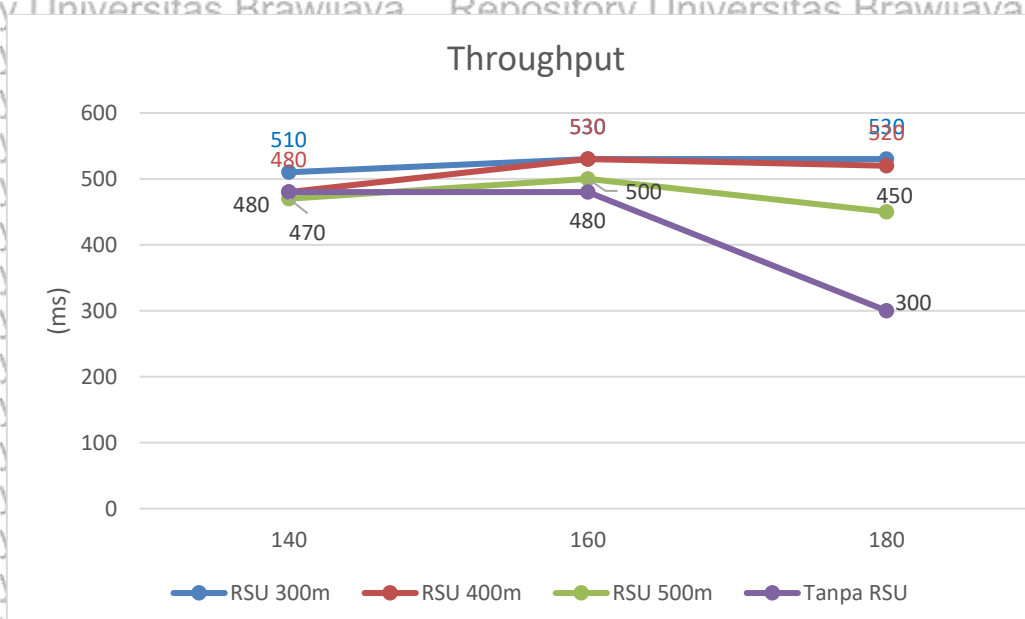
5.1.1 *Throughput*

Pada pengujian nilai *throughput*, nilai yang diambil didapatkan dari hasil simulasi dengan jarak RSU berdasarkan jumlah *node* kendaraan 140, 160, dan 180. Hasil yang didapat dari pengujian *Throughput* bisa dilihat pada Tabel 5.1.

Tabel 5.1 Hasil Pengujian nilai *throughput*

Jumlah Kendaraan	RSU 300m (bps)	RSU 400m (bps)	RSU 500m (bps)	Tanpa RSU (bps)
140 Kendaraan	510	480	470	480
160 Kendaraan	530	530	500	480
180 Kendaraan	530	520	450	300

Berdasarkan data diatas, dapat ditampilkan grafik yang mempresentasikan hasil dari ketiga hasil perhitungan yang ditunjukkan pada Gambar 5.2.



Gambar 5.2 Grafik *throughput* pengujian kepadatan kendaraan

Berdasarkan grafik data di atas, diperoleh hasil nilai uji dari skenario kepadatan kendaraan dengan parameter *throughput*, dimana skenario yang mempunyai RSU lebih baik daripada yang tidak mempunyai RSU. Pada jarak tiap RSU 300m didapatkan hasil *throughput* terbaik dengan semua nilai berada di atas 500 bps. Terdapat peningkatan nilai dengan *node* kendaraan 140, dan 160 dengan jarak RSU tiap 300m dan 400m dikarenakan banyaknya jumlah *node* dan RSU yang diletakkan dapat berpengaruh terhadap *throughput* seperti hasil yang sudah didapat karena semakin banyak *node* maka semakin besar kemungkinan *node* tujuan ditemukan. Jika penempatan tiap RSU semakin menjauh, maka nilai *throughput* nya cenderung akan menurun karena RSU dapat menambahkan kinerja routing pada VANET, maka jika semakin tinggi kinernya maka akan semakin baik nilai pada *throughput*. Begitu pula dengan jumlah *node*, semakin banyak jumlah *node* maka nilai *throughput* cenderung meningkat kecuali pada jumlah *node* 180 dimana jarak tiap RSU adalah 500m dan tanpa menggunakan RSU hal ini diduga karena adanya karakteristik VANET yaitu mobilitas kendaraan yang berubah-ubah mengakibatkan nilainya mengalami penurunan.

Pada kendaraan berjumlah 180, nilai *throughput* pada kendaraan yang tidak menggunakan RSU turun jauh. Hal ini terjadi dikarenakan karena karakteristik VANET yang mempunyai mobilitas tinggi sehingga posisi kendaraan sulit ditentukan atau kendaraan tersebut tidak mempunyai RSU sebagai *node* tambahan yang berfungsi untuk menambahkan nilai *throughput* dan juga hanya melalui sedikit *node* sehingga nilai *throughput* nya tidak sebaik yang melalui banyak *node* kendaraan serta *node* tambahan yaitu RSU. Berdasarkan Gambar 5.3 *node* sumber sampai ke *node* tujuan terdapat 70 *event* atau 70 kejadian dalam penyampaian data, sedangkan *trace* file pada gambar 6.4 hanya menunjukkan 50 *event* atau 50 kejadian dalam penyampaian data. Pada gambar 5.3 terdapat RSU

sebagai node pengirim yang mempunyai dengan node nomer 187 yang ditunjukkan pada baris ke 28706, dimana node RSU ini sebagai penambah event, sebagaimana semakin banyak kendaraan atau node yang berfungsi sebagai penerus data atau informasi maka semakin baik nilai *thorughput* nya.

```

23715 s 12.159952786 12 AGT --- 1129 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [5] 0 0
23716 r 12.159952786 12 RTR --- 1129 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [5] 0 0
23717 s 12.159952786 12 RTR --- 1129 cbr 603 [0 0 0 0] ----- [12:0 11:0 31 11] [5] 0 0
23718 r 12.160427706 12 MAC --- 0 RTS 44 [1726 b c 0]
23719 s 12.160780075 11 MAC --- 0 RTS 44 [1726 b c 0]
23720 r 12.160790075 11 MAC --- 0 CTS 38 [15ec c 0 0]
23721 r 12.161094364 12 MAC --- 0 CTS 38 [15ec c 0 0]
23722 s 12.161104364 12 MAC --- 1129 cbr 661 [13a b c 800] ----- [12:0 11:0 31 11] [5] 0 0
23723 r 12.161367203 111 RTR --- 1130 gpsr 29 [0 0 0 0] ----- [111:255 -1:255 32 0]
23724 s 12.162177738 79 RTR --- 1131 gpsr 29 [0 0 0 0] ----- [79:255 -1:255 32 0]
23725 s 12.162532311 37 RTR --- 1132 gpsr 29 [0 0 0 0] ----- [37:255 -1:255 32 0]
23726 s 12.163127311 37 MAC --- 1132 gpsr 87 [0 ffffffff 25 800] ----- [37:255 -1:255 32 0]
23727 r 12.163400647 165 RTR --- 1133 gpsr 29 [0 0 0 0] ----- [165:255 -1:255 32 0]
23728 s 12.163555647 165 MAC --- 1133 gpsr 87 [0 ffffffff a5 800] ----- [165:255 -1:255 32 0]
23729 D 12.163556090 127 MAC COL 1133 gpsr 87 [0 ffffffff a5 800] ----- [165:255 -1:255 32 0]
23730 D 12.163556261 91 MAC COL 1133 gpsr 87 [0 ffffffff a5 800] ----- [165:255 -1:255 32 0]
23731 D 12.163556273 70 MAC COL 1133 gpsr 87 [0 ffffffff a5 800] ----- [165:255 -1:255 32 0]
23732 D 12.163556341 93 MAC COL 1133 gpsr 87 [0 ffffffff a5 800] ----- [165:255 -1:255 32 0]
23733 D 12.163556368 164 MAC COL 1133 gpsr 87 [0 ffffffff a5 800] ----- [165:255 -1:255 32 0]
23734 r 12.163823759 15 MAC --- 1132 gpsr 29 [0 ffffffff 25 800] ----- [37:255 -1:255 32 0]
23735 r 12.163824104 95 MAC --- 1132 gpsr 29 [0 ffffffff 25 800] ----- [37:255 -1:255 32 0]
23736 r 12.163824124 132 MAC --- 1132 gpsr 29 [0 ffffffff 25 800] ----- [37:255 -1:255 32 0]
23737 r 12.163848759 15 RTR --- 1132 gpsr 29 [0 ffffffff 25 800] ----- [37:255 -1:255 32 0]
23765 r 12.166417653 11 AGT --- 1129 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [5] 1 0

```

Gambar 5.3 Trace File *Throughput* Kendaraan 180 dengan RSU 300m

```

28698 s 12.360814600 12 AGT --- 1290 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [6] 0 0
28699 r 12.360814600 12 RTR --- 1290 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [6] 0 0
28700 s 12.360814600 12 RTR --- 1290 cbr 603 [0 0 0 0] ----- [12:0 11:0 31 11] [6] 0 0
28701 s 12.361349600 12 MAC --- 0 RTS 44 [1726 b c 0]
28702 r 12.361701889 11 MAC --- 0 RTS 44 [1726 b c 0]
28703 s 12.361711889 11 MAC --- 0 CTS 38 [15ec c 0 0]
28704 r 12.362016178 12 MAC --- 0 CTS 38 [15ec c 0 0]
28705 s 12.362026178 12 MAC --- 1290 cbr 661 [13a b c 800] ----- [12:0 11:0 31 11] [6] 0 0
28706 s 12.362770016 187 RTR --- 1291 gpsr 29 [0 0 0 0] ----- [187:255 -1:255 32 0]
28707 s 12.363156901 67 RTR --- 1292 gpsr 29 [0 0 0 0] ----- [67:255 -1:255 32 0]
28708 s 12.363325016 187 MAC --- 1291 gpsr 87 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28709 r 12.364021214 90 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28710 r 12.364021429 145 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28711 r 12.364021504 96 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28712 r 12.364021592 22 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28713 r 12.364021718 119 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28714 r 12.364021752 163 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28715 r 12.364021798 98 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28716 r 12.364021844 149 MAC --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28717 r 12.364046214 90 RTR --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28718 r 12.364046429 145 RTR --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28719 r 12.364046504 96 RTR --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28720 r 12.364046592 22 RTR --- 1291 gpsr 29 [0 ffffffff bb 800] ----- [187:255 -1:255 32 0]
28768 r 12.367339467 11 AGT --- 1290 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [6] 1 0

```

Gambar 5.4 Trace File *Throguhput* Kendaraan 180 Tanpa RSU

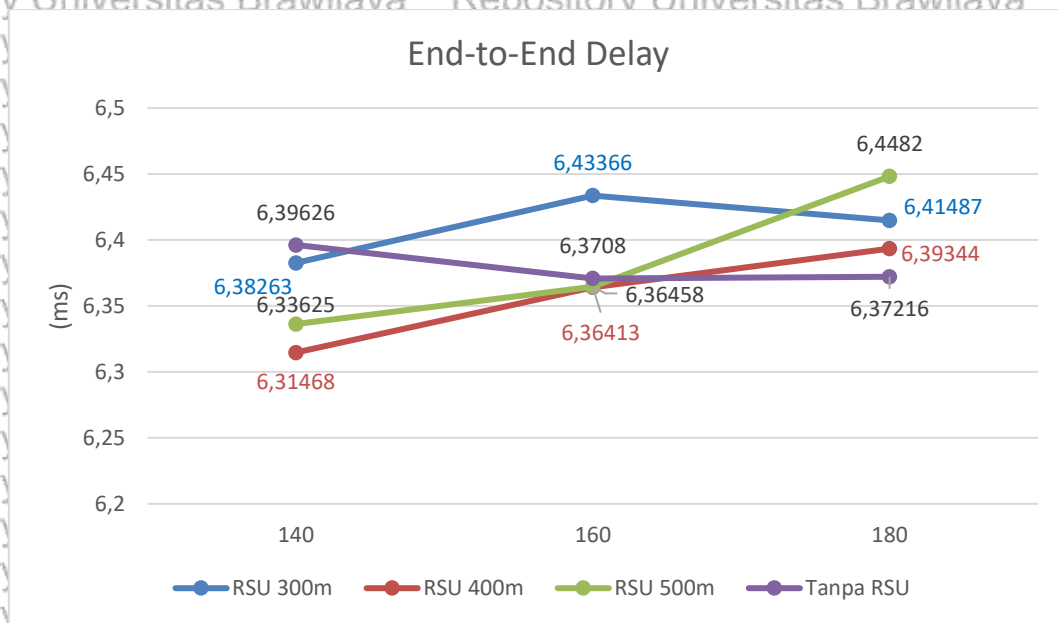
5.1.2 End-to-end Delay

Pada pengujian nilai *end-to-end delay*, nilai yang diambil didapatkan dari hasil simulasi dengan jarak RSU berdasarkan jumlah node kendaraan 140, 160, dan 180. Hasil yang didapat dari pengujian *End-to-end Delay* bisa dilihat pada Tabel 5.2.

Tabel 5.2 Hasil Pengujian nilai *end-to-end delay*

Jumlah Kendaraan	RSU 300m (m/s)	RSU 400m (m/s)	RSU 500m (m/s)	Tanpa RSU (m/s)
140 Kendaraan	6,38263	6,31468	6,33625	6,39626
160 Kendaraan	6,43366	6,36413	6,36458	6,3708
180 Kendaraan	6,41487	6,39344	6,4482	6,37216

Berdasarkan data diatas, dapat ditampilkan grafik yang mempresentasikan hasil dari ketiga hasil perhitungan yang ditunjukkan pada Gambar 5.5.



Gambar 5.5 Grafik *end-to-end delay* pengujian kepadatan kendaraan

Berdasarkan grafik data diatas, diperoleh hasil nilai uji dari skenario kepadatan kendaraan dengan parameter *end-to-end delay*. Pada jumlah node 140 terhadap peneletakan tiap RSU 400m didapatkan nilai *end-to-end delay* lebih baik dibandingkan RSU tiap 300m dan 500m yaitu sebesar 6,31468m/s. GPRS mencari rute dengan selalu mengirim sinyal ke tetangga terdekatnya yang menyebabkan semakin banyak node maka delaynya akan semakin tinggi namun pada skenario pada jumlah *node* 180 yang memakai RSU 300m mengalami penurunan *delay*. Hal ini terjadi karena pada skenario tersebut tidak terlalu banyak memakan waktu dalam proses permintaan informasi ke *node* tujuan, sebagai contoh pada gambar 5.6 merupakan *trace file* dengan jumlah *node* 180 dengan memakai RSU tiap 300m dan Gambar 5.7 sebagai pembandingan merupakan *trace file* dengan jumlah *node* 180 dengan memakai RSU tiap 500m.

```

4406 s 1.0000000000 12 AGT --- 191 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [0] 0 0
4407 r 1.0000000000 12 RTR --- 191 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [0] 0 0
4408 s 1.0000000000 12 RTR --- 191 gpsr 69 [0 0 0 0] ----- [12:255 11:255 32 0]
4409 M 1.00000 0 (1216.95, 689.92, 0.00), (1226.03, 685.32), 10.19
4410 M 1.00000 1 (2460.98, 1214.80, 0.00), (2460.98, 1214.80), 13.89
4411 s 1.000535000 12 MAC --- 191 gpsr 127 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4412 r 1.001551248 24 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4413 r 1.001551289 11 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4414 r 1.001551311 114 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4415 r 1.001551481 124 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4416 r 1.001551482 186 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4417 r 1.001551500 131 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4418 r 1.001551630 79 MAC --- 191 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
8744 r 2.499098310 30 RTR --- 384 gpsr 29 [0 ffffffff 0 800] ----- [0:255 -1:255 32 0]
8745 s 2.799624426 12 AGT --- 385 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [1] 0 0

```

Gambar 5.6 Trace File *end to end delay* Kendaraan 180 RSU 300m


```

4245 s 1.0000000000 _12_ AGT --- 188 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [0] 0 0
4246 r 1.0000000000 _12_ RTR --- 188 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [0] 0 0
4247 s 1.0000000000 _12_ RTR --- 188 gpsr 69 [0 0 0 0] ----- [12:255 11:255 32 0]
4248 M 1.00000 0 (1216.95, 689.92, 0.00), (1226.03, 685.32), 10.19
4249 M 1.00000 1 (2460.98, 1214.80, 0.00), (2460.98, 1214.80), 13.89
4250 s 1.000115000 _12_ MAC --- 188 gpsr 127 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4251 r 1.001131248 _24_ MAC --- 188 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4252 r 1.001131289 _11_ MAC --- 188 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4253 r 1.001131311 _114_ MAC --- 188 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4254 r 1.001131481 _124_ MAC --- 188 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4255 r 1.001131500 _131_ MAC --- 188 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
4256 r 1.001131630 _79_ MAC --- 188 gpsr 69 [0 ffffffff c 800] ----- [12:255 11:255 32 0]
8421 r 2.498678310 _30_ RTR --- 378 gpsr 29 [0 ffffffff 0 800] ----- [0:255 -1:255 32 0]
8422 M 3.00000 0 (1231.77, 682.41, 0.00), (1239.33, 680.13), 8.07
8423 M 3.00000 1 (2448.88, 1219.87, 0.00), (2436.60, 1225.02), 13.31
8424 M 3.00000 2 (2948.75, 439.73, 0.00), (2928.68, 447.47), 21.51
8425 M 3.00000 3 (2577.23, 943.29, 0.00), (2577.23, 943.29), 13.89
8426 s 3.993343664 _12_ AGT --- 379 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [1] 0 0

```

Gambar 5.7 Trace File *end to end delay* Kendaraan 180 RSU 500m

Pada Gambar 5.5 dan Gambar 5.6 mempunyai *start* yang sama pada *node* sumber merequest informasi, tetapi pada Gambar 5.6 terdapat suatu kejadian yang membuat *node* sumber mengirim kembali permintaan informasinya dengan memakan banyak waktu daripada skenario yang terdapat pada RSU tiap 300m. Hal ini yang membuat nilai *end to end delay* dari skenario *node* kendaraan 180 dengan RSU 300m lebih baik daripada skenario dari nilai *end to end delay* dari *node* 180 dengan RSU 500m.

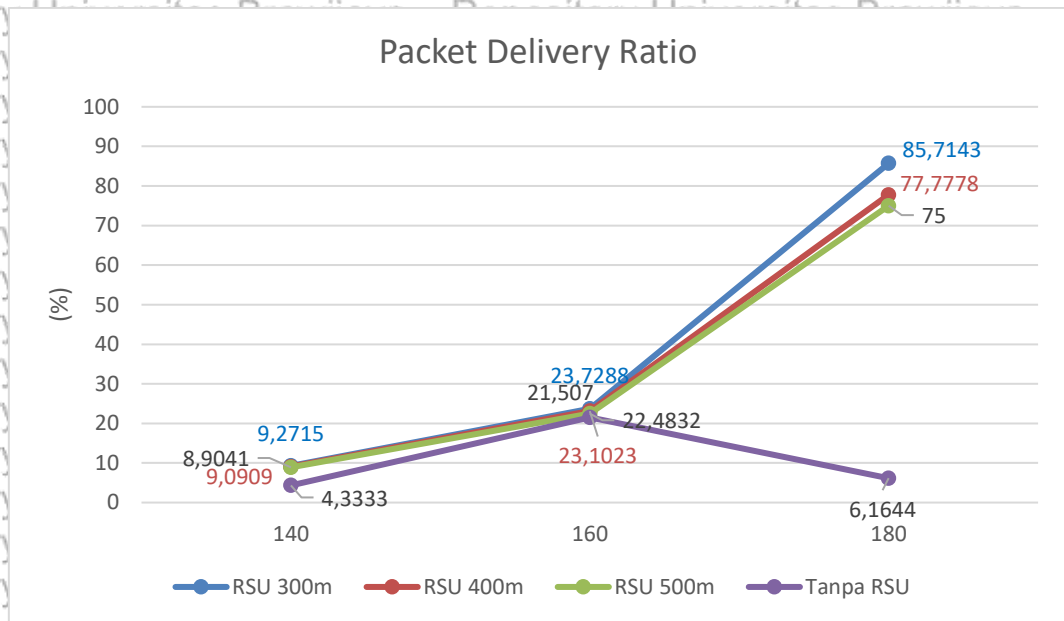
5.1.3 Packet Delivery Ratio

Pada pengujian nilai *Packet delivery ratio*, nilai yang diambil didapatkan dari hasil simulasi dengan jarak RSU berdasarkan jumlah *node* kendaraan 140, 160, dan 180. Hasil yang didapat dari pengujian *Packet delivery ratio* bisa dilihat pada Tabel 5.3.

Tabel 5.3 Hasil Pengujian nilai *packet delivery ratio*

Jumlah Kendaraan	RSU 300m (%)	RSU 400m (%)	RSU 500m (%)	Tanpa RSU (%)
140 Kendaraan	9,2715	9,0909	8,9041	4,3333
160 Kendaraan	23,7288	23,1023	22,4832	21,507
180 Kendaraan	85,7143	77,7778	75	6,1644

Berdasarkan data diatas, dapat ditampilkan grafik yang mempresentasikan hasil dari ketiga hasil perhitungan yang ditunjukkan pada Gambar 5.7.



Gambar 5.8 Grafik *packet delivery ratio* pengujian kepadatan kendaraan

Berdasarkan grafik data diatas, diperoleh hasil nilai uji dari skenario kepadatan kendaraan dengan parameter *packet delivery ratio*. Pada jumlah *node* 180 dengan penempatan RSU tiap 300m didapatkan nilai *packet delivery ratio* terbaik sebesar 85,7143%. Pada saat kendaraan berjumlah 180 dengan jarak tiap RSU 300, 400, dan 500m mengalami peningkatan. Nilai *packet delivery ratio* meningkat seiring pada jumlah *node* bertambah, serta nilai *packet delivery ratio* berkurang seiring jarak tiap RSU diperlebar. Hal ini disebabkan karena RSU berfungsi sebagai *node* yang diam sehingga memungkinkan peningkatan nilai pada *packet delivery ratio*, sehingga makin banyak RSU maka nilai *packet delivery ratio* akan meningkat.

Routing GPSR mempunyai 2 cara yaitu *greedy forwarding* dan *perimeter forwarding*. Pada saat protokol GPSR memakai *freedy forwarding* maka jalur dari *node* sumber ke *node* tujuan sudah didapatkan sehingga paket data pasti sampai ke *node* tujuan jika tidak ada masalah, jika protokol GPSR menggunakan *perimeter forwarding* maka *node* sumber belum tentu berhasil mendapatkan jalur ke *node* tujuan sehingga *node* sumber terus mengirimkan permintaan tanpa ada kepastian bahwa *node* sumber berhasil mendapatkan jalur ke *node* tujuan sehingga menyebabkan nilai *packet delivery ratio* bernilai rendah. Dalam hal ini RSU berperan sebagai pembantu agar *node* sumber dapat menemukan lajur pengiriman paket data ke *node* tujuan tanpa harus mengulang proses *greedy perimeter* yang menyebabkan penurunan nilai pada *packet delivery ratio*.

Dalam skenario dengan *node* 180 tanpa menggunakan RSU terdapat penurunan nilai dikarenakan pengiriman permintaan informasi dari *node* sumber terus dilakukan walau sudah tidak mendapat balasan lagi dari *node* tujuan. Berikut potongan contoh *trace file* :


```

Search "AGT" (13 hits in 1 file)
C:\Skripsi\Skripsi 3\trace\300\180.tr (13 hits)
Line 4406: s 1.000000000 12 AGT --- 191 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [0] 0 0
Line 8745: s 2.799624426 12 AGT --- 385 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [1] 0 0
Line 8755: r 2.805889292 11 AGT --- 385 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [1] 1 0
Line 8761: s 3.984529086 12 AGT --- 386 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [2] 0 0
Line 8771: r 3.990913952 11 AGT --- 386 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [2] 1 0
Line 17188: s 6.550932940 12 AGT --- 769 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [3] 0 0
Line 17198: r 6.557597806 11 AGT --- 769 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [3] 1 0
Line 20212: s 8.351978645 12 AGT --- 906 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [4] 0 0
Line 20222: r 8.358443511 11 AGT --- 906 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [4] 1 0
Line 25654: s 10.809297219 12 AGT --- 1153 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [5] 0 0
Line 25664: r 10.815482086 11 AGT --- 1153 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [5] 1 0
Line 28698: s 12.360814600 12 AGT --- 1290 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [6] 0 0
Line 28768: r 12.367339467 11 AGT --- 1290 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [6] 1 0

```

Gambar 5.9 Trace File Packet Delivery Ratio kendaraan 180 RSU 300m

```

Search "agt" (310 hits in 1 file)
C:\Skripsi\Skripsi 3\trace\tidak\180.tr (310 hits)
Line 3868: s 1.000000000 12 AGT --- 178 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [0] 0 0
Line 7683: s 2.626265757 12 AGT --- 359 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [1] 0 0
Line 7693: r 2.632450624 11 AGT --- 359 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [1] 1 0
Line 11370: s 4.495842935 12 AGT --- 537 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [2] 0 0
Line 11396: r 4.502327801 11 AGT --- 537 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [2] 1 0
Line 15104: s 6.695928876 12 AGT --- 717 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [3] 0 0
Line 15114: r 6.702333742 11 AGT --- 717 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [3] 1 0
Line 18819: s 9.528107157 12 AGT --- 896 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [4] 0 0
Line 18829: r 9.534632023 11 AGT --- 896 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [4] 1 0
Line 23715: s 12.159952786 12 AGT --- 1129 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [5] 0 0
Line 23765: r 12.166417653 11 AGT --- 1129 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [5] 1 0
Line 30033: s 14.844871633 12 AGT --- 1432 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [6] 0 0
Line 30043: r 14.851436522 11 AGT --- 1432 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [6] 1 0
Line 121369: r 61.188867561 11 AGT --- 5548 cbr 603 [13a b c 800] ----- [12:0 11:0 31 11] [28] 1 0
Line 125536: s 63.126372833 12 AGT --- 5727 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [29] 0 0
Line 129648: s 64.544707663 12 AGT --- 5906 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [30] 0 0
Line 130349: s 66.068306020 12 AGT --- 5932 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [31] 0 0
Line 137104: s 68.398738377 12 AGT --- 6225 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [32] 0 0
Line 141018: s 70.353989113 12 AGT --- 6389 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [33] 0 0
Line 144273: s 72.240858630 12 AGT --- 6524 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [34] 0 0
Line 146648: s 73.411737530 12 AGT --- 6623 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [35] 0 0
Line 151071: s 75.548416041 12 AGT --- 6802 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [36] 0 0
Line 155583: s 77.678990665 12 AGT --- 6981 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [37] 0 0
Line 160474: s 80.037090034 12 AGT --- 7172 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [38] 0 0
Line 165918: s 82.148970215 12 AGT --- 7369 cbr 512 [0 0 0 0] ----- [12:0 11:0 32 0] [39] 0 0

```

Gambar 5.10 Trace File Packet Delivery Ratio Kendaraan 180 Tanpa RSU

Pada Gambar 5.9 *node* terdapat 13 kejadian dalam permintaan serta pengiriman informasi, sedangkan pada Gambar 6.10 terdapat 310 kejadian dan kebanyakan kejadian tersebut hanya *node* sumber yang terus meminta informasi saja dimana *node* tujuan sudah tidak melakukan aktifitasnya lagi. Hal ini menyebabkan nilai dari *packet delivery ratio* pada skenario kendaraan 180 dengan tanpa menggunakan RSU menurun jauh karena tidak ada pesannya yang sampai ke tujuan.

5.2 Pengujian Skenario Kecepatan Kendaraan

Dalam sub bab ini akan menjelaskan hasil uji dari skenario kinerja GPSR menggunakan RSU berdasarkan kecepatan kendaraan yang kecepatannya adalah 40 km/jam, 50km/jam, dan 60km/jam dengan RSU yang topologinya mengacu ke Gambar 3.6, Gambar 3.7, dan Gambar 3.8. Karakteristik VANET network topology yaitu karena tingginya suatu mobilitas node maka kecepatan node sering berubah dan menyebabkan topologi jaringan yang ada di VANET berubah juga. Maka diperlukan skenario yang menggunakan kecepatan *node* untuk menentukan kualitas jaringan beberapa kecepatan agar didapat data yang valid.



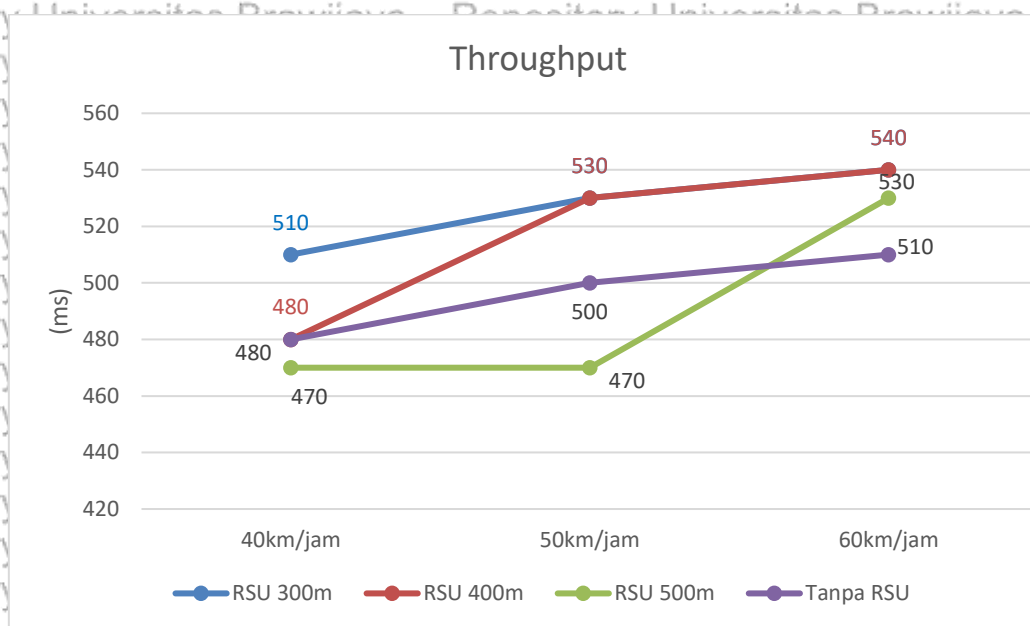
5.2.1 Throughput

Pada pengujian nilai *throughput*, nilai yang diambil didapatkan dari hasil simulasi dengan jarak RSU berdasarkan kecepatan kendaraan sebesar 40km/jam, 50km/jam, dan 60km/jam. Hasil yang didapat dari pengujian *throughput* bisa dilihat pada Tabel 5.4.

Tabel 5.4 Hasil Pengujian nilai *throughput*

Kecepatan Kendaraan	RSU 300m (bps)	RSU 400m (bps)	RSU 500m (bps)	Tanpa RSU (%)
40km/jam	510	480	470	480
50km/jam	530	530	470	500
60km/jam	540	540	530	510

Berdasarkan data yang diperoleh Tabel 5.4, maka dapat dibuat grafik hasil pengujian nilai *throughput* pada Gambar 5.11 sebagai berikut.



Gambar 5.11 Grafik *Throughput* Pengujian Kecepatan Kendaraan

Berdasarkan grafik data diatas, diperoleh hasil nilai uji dari skenario jarak tiap RSU dengan parameter *throughput*. Pada jarak tiap RSU adalah 300 m didapatkan nilai *throughput* terbaik dibandingkan RSU pada tiap 400m, 500m, dan tanpa memakai RSU. Hal ini dikarenakan semakin cepat kendaraan atau node bergerak maka waktu rata-rata yang diterima akan semakin besar. Kecepatan kendaraan mempunyai pengaruh terhadap routing GPSR terutama *perimeter forwarding* karena semakin cepat kendaraan maka *node* kendaraan semakin banyak bertemu



node lain dimana pengiriman permintaan dari *node* sumber akan semakin banyak dimana nilai *throughput* akan semakin besar jika banyak *node* yang terlibat. Hasil *throughput* pada *node* tanpa menggunakan RSU mempunyai nilai rata rata paling kecil karena pada skenario tersebut tidak mempunyai RSU yang berlaku sebagai *node* tambahan yang tidak bergerak, dimana jika semakin banyak *node* yang terhubung maka semakin besar nilai dari *throughput*. Pada skenario RSU 500m dengan kecepatan 50km/jam didapatkan nilai *throughput* yang kecil dibandingkan dengan skenario lain, berikut *trace file* dan penjelasannya :

```

3198 s 2.152996500 1 AGT --- 189 cbr 512 [0 0 0 0] ----- [1:0 5:0 32 0] [1] 0 0
3199 r 2.152996500 1 RTR --- 189 cbr 512 [0 0 0 0] ----- [1:0 5:0 32 0] [1] 0 0
3200 s 2.152996500 1 RTR --- 189 cbr 603 [0 0 0 0] ----- [1:0 5:0 31 5] [1] 0 0
3201 s 2.153311500 1 MAC --- 0 RTS 44 [1726 5 1 0]
3202 r 2.153663778 5 MAC --- 0 RTS 44 [1726 5 1 0]
3203 s 2.153673778 5 MAC --- 0 CTS 38 [15ec 1 0 0]
3204 r 2.153978055 1 MAC --- 0 CTS 38 [15ec 1 0 0]
3205 s 2.153988055 1 MAC --- 189 cbr 661 [13a 5 1 800] ----- [1:0 5:0 31 5] [1] 0 0
3206 s 2.155690870 52 RTR --- 190 gpsr 29 [0 0 0 0] ----- [52:255 -1:255 32 0]
3207 s 2.155925870 52 MAC --- 190 gpsr 87 [0 ffffffff 34 800] ----- [52:255 -1:255 32 0]
3208 r 2.156622185 2 MAC --- 190 gpsr 29 [0 ffffffff 34 800] ----- [52:255 -1:255 32 0]
3209 r 2.156622307 53 MAC --- 190 gpsr 29 [0 ffffffff 34 800] ----- [52:255 -1:255 32 0]
3210 r 2.156622438 91 MAC --- 190 gpsr 29 [0 ffffffff 34 800] ----- [52:255 -1:255 32 0]
3226 r 2.159301332 5 AGT --- 189 cbr 603 [13a 5 1 800] ----- [1:0 5:0 31 5] [1] 1 0

```

Gambar 5.12 Trace File *Throughput* kecepatan kendaraan 50km/jam RSU 500m

```

4654 s 2.407205303 1 AGT --- 272 cbr 512 [0 0 0 0] ----- [1:0 5:0 32 0] [1] 0 0
4655 r 2.407205303 1 RTR --- 272 cbr 512 [0 0 0 0] ----- [1:0 5:0 32 0] [1] 0 0
4656 s 2.407205303 1 RTR --- 272 cbr 603 [0 0 0 0] ----- [1:0 5:0 31 5] [1] 0 0
4657 s 2.407840303 1 MAC --- 0 RTS 44 [1726 5 1 0]
4658 s 2.407911580 56 RTR --- 273 gpsr 29 [0 0 0 0] ----- [56:255 -1:255 32 0]
4659 s 2.407986580 56 MAC --- 273 gpsr 87 [0 ffffffff 38 800] ----- [56:255 -1:255 32 0]
4660 r 2.408192571 5 MAC --- 0 RTS 44 [1726 5 1 0]
4661 s 2.408202571 5 MAC --- 0 CTS 38 [15ec 1 0 0]
4662 r 2.408506840 1 MAC --- 0 CTS 38 [15ec 1 0 0]
4663 s 2.408516840 1 MAC --- 272 cbr 661 [13a 5 1 800] ----- [1:0 5:0 31 5] [1] 0 0
4664 r 2.408682953 32 MAC --- 273 gpsr 29 [0 ffffffff 38 800] ----- [56:255 -1:255 32 0]
4665 r 2.408682980 140 MAC --- 273 gpsr 29 [0 ffffffff 38 800] ----- [56:255 -1:255 32 0]
4666 r 2.408682995 80 MAC --- 273 gpsr 29 [0 ffffffff 38 800] ----- [56:255 -1:255 32 0]
4667 r 2.408683079 35 MAC --- 273 gpsr 29 [0 ffffffff 38 800] ----- [56:255 -1:255 32 0]
4688 r 2.413830108 5 AGT --- 272 cbr 603 [13a 5 1 800] ----- [1:0 5:0 31 5] [1] 1 0

```

Gambar 5.13 Trace File *Throughput* kecepatan kendaraan 50km/jam RSU 300m

Pada Gambar 5.12 dan Gambar 5.13 memperlihatkan potongan *trace file* yang hasil uji yang didapat. *Throughput* pada skenario kendaraan 50km/jam dengan RSU 500m didapatkan hasil paling kecil karena pada skenario tersebut jumlah kejadian atau *event* yang terjadi komunikasi antar *node* sedikit sesuai gambar diatas. Pada skenario tersebut *node* pengirim mengirimkan *request* pada *event* ke 3198 sedangkan pada RSU 300m *node* pengirim mengirimkan *request* pada *event* ke 4654. Lalu untuk bisa menyampaikan informasi, skenario dengan RSU 300m lebih banyak membutuhkan *event* atau melibatkan lebih banyak *node* untuk bisa mengirimkan informasi ke tujuan dimana dalam RSU 300m terjadi 34 *event* sedangkan RSU 500m hanya terjadi 28 *event* yang dibutuhkan pesan terkirim dari *node* pengirim ke *node* tujuan. Hal ini membuktikan bahwa semakin banyak *node* yang terlibat dalam proses pengiriman data maka semakin baik nilai *throughput*.



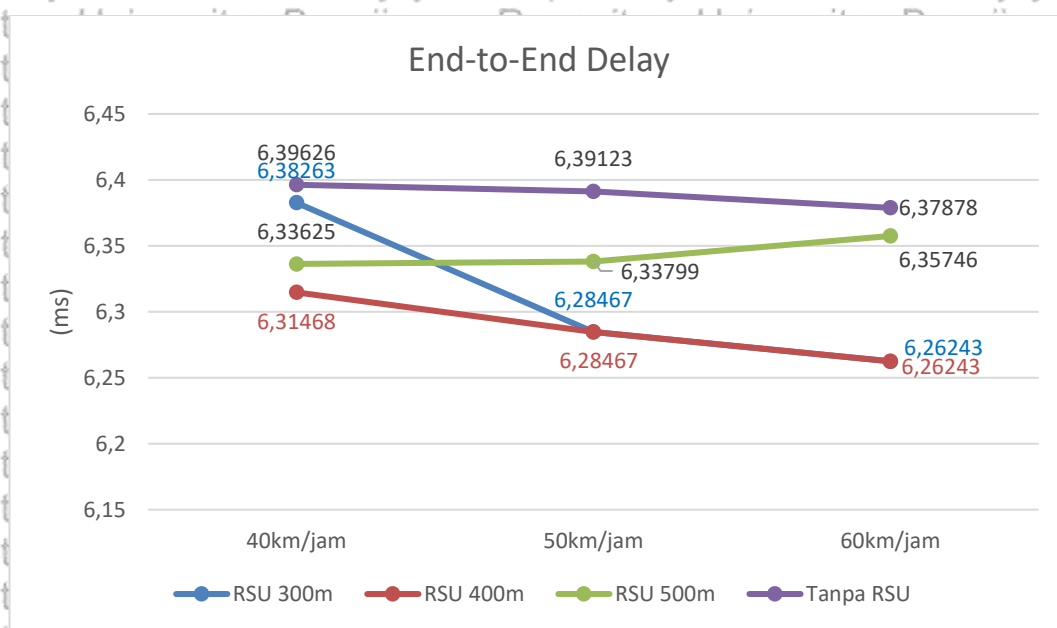
5.2.2 End-to-end Delay

Pada pengujian nilai *end-to-end delay*, nilai yang diambil didapatkan dari hasil simulasi dengan jarak RSU berdasarkan kecepatan kendaraan sebesar 40km/jam, 50km/jam, dan 60km/jam. Hasil yang didapat dari pengujian *end-to-end delay* bisa dilihat pada Tabel 6.5.

Tabel 5.5 Hasil Pengujian nilai *end-to-end delay*

Kecepatan Kendaraan	RSU 300m (m/s)	RSU 400m (m/s)	RSU 500m (m/s)	Tanpa RSU (m/s)
40km/jam	6,38263	6,31468	6,33625	6,39626
50km/jam	6,28467	6,28467	6,33799	6,39123
60km/jam	6,26243	6,26243	6,35746	6,37878

Berdasarkan data yang diperoleh Tabel 5.5, maka dapat dibuat grafik hasil pengujian nilai *End-to-end delay* pada Gambar 5.12 sebagai berikut.



Gambar 5.14 Grafik *End-to-end delay* Pengujian Kecepatan Kendaraan

Berdasarkan grafik data diatas, diperoleh hasil nilai uji dari skenario jarak tiap RSU dengan parameter *end-to-end delay*. Pada jarak tiap RSU adalah tiap 400m didapatkan nilai *end-to-end delay* terbaik dibandingkan pada skenario jarak tiap RSU 300m, 500m, dan tanpa menggunakan RSU. Pada skenario kecepatan 50 km/jam dan kecepatan 60 km/jam nilai *end to end delay* sebagian besar menurun. Hal ini disebabkan semakin cepat *node* berjalan maka semakin cepat *node* bertemu dengan satu sama lain yang membuat informasi sampai ke *node* tujuan sampa lebih cepat dan begitu pula pengiriman informasi kembali ke *node* sumbernya. Pada skenario RSU 500m nilai pada *end to-end delay* terus mengalami kenaikan, berikut *trace file* serta penjelasan dari salah satu contoh skenario :



```
2683 s 1.000000000 1 AGT --- 143 cbr 512 [0 0 0 0] ----- [1:0 3:0 32 0] [0] 0 0
2684 r 1.000000000 1 RTR --- 143 cbr 512 [0 0 0 0] ----- [1:0 3:0 32 0] [0] 0 0
2685 s 1.000000000 1 RTR --- 143 gpsr 69 [0 0 0 0] ----- [1:255 3:255 32 0]
2686 M 1.00000 0 (958.59, 580.59, 0.00), (964.16, 587.92), 9.21
2687 M 1.00000 1 (1990.05, 426.45, 0.00), (1990.05, 426.45), 13.89
2688 s 1.000375000 1 MAC --- 143 gpsr 127 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2689 r 1.001391060 49 MAC --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2690 r 1.001391210 3 MAC --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2691 r 1.001391358 19 MAC --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2692 r 1.001391372 113 MAC --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2693 r 1.001391373 41 MAC --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2694 r 1.001391481 67 MAC --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2695 r 1.001416060 49 RTR --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2696 f 1.001416060 49 RTR --- 143 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
3790 r 2.201990704 50 RTR --- 200 gpsr 29 [0 ffffffff 6b 800] ----- [107:255 -1:255 32 0]
3791 r 2.202350771 3 MAC --- 198 cbr 603 [13a 3 1 800] ----- [1:0 3:0 31 3] [1] 1 0
3792 s 2.202360771 3 MAC --- 0 ACK 38 [0 1 0 0]
3793 r 2.202375771 3 AGT --- 198 cbr 603 [13a 3 1 800] ----- [1:0 3:0 31 3] [1] 1 0
```

Gambar 5.15 Trace File *end-to-end delay* kecepatan kendaraan 60km/jam RSU 300m

```
2901 s 1.000000000 1 AGT --- 148 cbr 512 [0 0 0 0] ----- [1:0 3:0 32 0] [0] 0 0
2902 r 1.000000000 1 RTR --- 148 cbr 512 [0 0 0 0] ----- [1:0 3:0 32 0] [0] 0 0
2903 s 1.000000000 1 RTR --- 148 gpsr 69 [0 0 0 0] ----- [1:255 3:255 32 0]
2904 M 1.00000 0 (958.59, 580.59, 0.00), (964.16, 587.92), 9.21
2905 M 1.00000 1 (1990.05, 426.45, 0.00), (1990.05, 426.45), 13.89
2906 s 1.000675000 1 MAC --- 148 gpsr 127 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2907 r 1.001691060 49 MAC --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2908 r 1.001691210 3 MAC --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2909 r 1.001691358 19 MAC --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2910 r 1.001691372 113 MAC --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2911 r 1.001691373 41 MAC --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2912 r 1.001691481 67 MAC --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2913 r 1.001716060 49 RTR --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
2914 f 1.001716060 49 RTR --- 148 gpsr 69 [0 ffffffff 1 800] ----- [1:255 3:255 32 0]
5758 r 2.613720112 3 MAC --- 299 cbr 603 [13a 3 1 800] ----- [1:0 3:0 31 3] [1] 1 0
5759 s 2.613730112 3 MAC --- 0 ACK 38 [0 1 0 0]
5760 r 2.613745112 3 AGT --- 299 cbr 603 [13a 3 1 800] ----- [1:0 3:0 31 3] [1] 1 0
```

Gambar 5.16 Trace File *end-to-end delay* kecepatan kendaraan 60km/jam RSU 500m

Pada Gambar 5.15 dan Gambar 5.16 mempunyai *start* yang sama pada *node* sumber merequest informasi, tetapi pada skenario RSU 500m dengan kecepatan 60km/jam waktu *node* mendapatkan paket data dari *node* sumber lebih lambat daripada skenario yang menggunakan RSU 300m dengan kecepatan 60km/jam. Hal ini bisa dilihat pada 2 gambar diatas dimana waktu *node* dengan skenario RSU 300m dengan kecepatan 60km/jam sampai pada tujuan adalah 2,202375771 detik sedangkan untuk skenario RSU 500m dkecepatan 60km/jam adalah 2,613745112 detik. Hal ini terjadi karena terdapat suatu kejadian yang membuat *node* yang membawa paket data dan meneruskannya ke *node* lain membutuhkan waktu lebih lama, hal ini dapat dilihat pada baris 2688 untuk skenario RSU 300m dan 2906 untuk skenario RSU 500m.



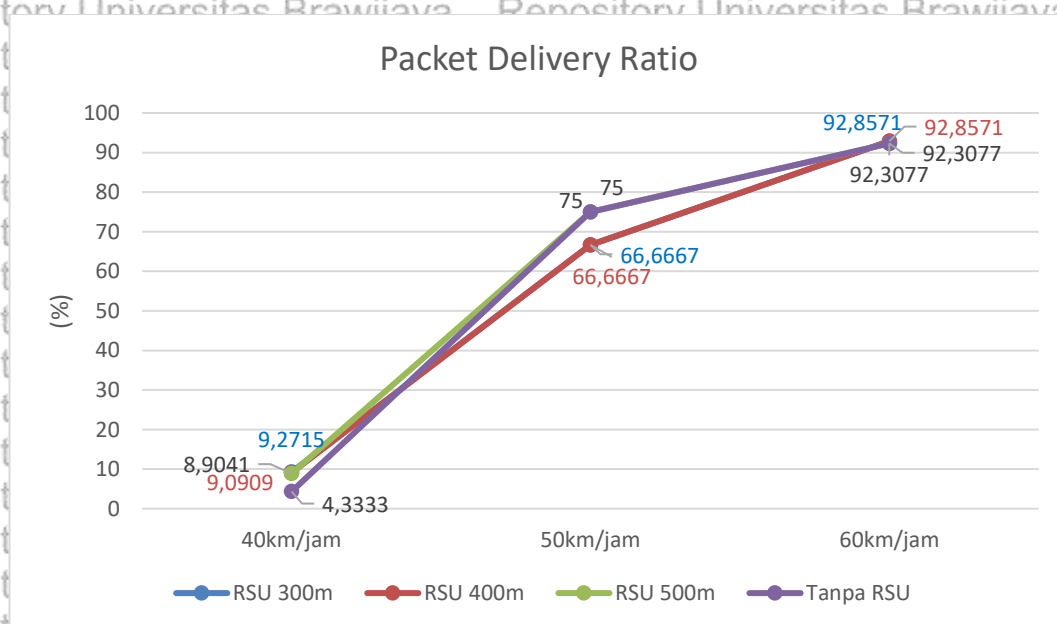
5.2.3 Packet Delivery Ratio

Pada pengujian nilai *packet delivery ratio*, nilai yang diambil didapatkan dari hasil simulasi dengan jarak RSU berdasarkan kecepatan kendaraan sebesar 40km/jam, 50km/jam, dan 60km/jam. Hasil yang didapat dari pengujian *packet delivery ratio* bisa dilihat pada Tabel 5.6.

Tabel 5.6 Hasil Pengujian nilai *packet delivery ratio*

Kecepatan Kendaraan	RSU 300m (%)	RSU 400m (%)	RSU 500m (%)	Tanpa RSU
40km/jam	9,2715	9,0909	8,9041	4,3333
50km/jam	66,6667	66,6667	75	75
60km/jam	92,8571	92,8571	92,3077	92,3077

Berdasarkan data yang diperoleh Tabel 5.6, maka dapat dibuat grafik hasil pengujian nilai *packet delivery ratio* pada Gambar 5.13 sebagai berikut.



Gambar 5.17 Grafik *Packet Delivery Ratio* Pengujian Kecepatan Kendaraan

Berdasarkan grafik data di atas, diperoleh hasil nilai uji dari skenario jarak tiap RSU dengan parameter *packet delivery ratio*. Pada jarak RSU adalah tiap 300m dan 400m dengan kecepatan 60km/jam didapatkan nilai tertinggi *packet delivery ratio* terbaik dibandingkan pada jarak RSU tiap 500m dan tanpa menggunakan RSU, karena RSU juga membantu untuk mengirimkan informasi sehingga nilai *packet delivery ratio* yang menggunakan RSU sebagian besar lebih baik daripada yang tidak menggunakan RSU. Pada grafik tersebut juga semakin besar kecepatan node maka semakin baik nilai dari *packet delivery ratio*, hal ini terjadi karena semakin



cepat suatu *node* maka *node* itu bisa lebih cepat bertemu dengan *node* lain dan menyampaikan informasi tersebut ke *node* lain sebelum paket tersebut dibuang karena tidak mempunyai *node* disekitar yang masuk dalam jangkauannya.



BAB 6 PENUTUP

6.1 Kesimpulan

Setelah didapatkan hasil dari uji coba, bisa didapatkan kesimpulan sebagai berikut :

1. Implementasi protokol GPSR pada VANET untuk komunikasi kendaraan dengan menggunakan RSU menggunakan NS2 sebagai simulator *traffic*, dan juga SUMO sebagai simulator pemodelan map. Dari hasil yang didapat, protokol GPSR pada VANET menggunakan RSU berpengaruh untuk meningkatkan kualitas komunikasi kendaraan yang parameter ujinya terdiri dari *throughput*, *end-to-end delay*, dan *packet delivery ratio*.
2. Berdasarkan hasil analisa yang didapat, hasil parameter QoS routing protokol GPSR dipengaruhi oleh adanya *node* RSU di jaringan VANET. Pada skenario kepadatan kendaraan, hasil pengujian pada nilai *thorughput* dan *packet delivery ratio* paling tinggi adalah pada saat *node* kendaraan berjumlah 180 dengan RSU berjarak tiap 300m yaitu dengan nilai 530 bps untuk *throughput* dan 85,7143% untuk nilai *packet delivery ratio*. Untuk nilai *end-to-end delay* nilai terbaik ada pada skenario *node* kendaraan yang berjumlah 140 kendaraan dengan RSU tiap 400m dengan nilai 6,31468 m/s. Untuk hasil pengujian dengan skenario kecepatan kendaraan, nilai *thorughput* terbaik juga terdapat di skenario kecepatan 60km/jam dengan RSU tiap 300m dengan nilai 540 bps. Untuk nilai *end-to-end delay* serta *packet delivery ratio* hasil terbaik didapatkan oleh skenario 60km/jam dengan jarak RSU 300m dan 400m dengan nilai yang sama yaitu 6,26243 m/s untuk niai *end-to-end delay* dan 92,8571% untuk nilai *packet delivery ratio*.

6.2 Saran

Saran yang dapat diberikan dari hasil penelitian *Studi Penempatan Roadside Unit (RSU) Pada Vehicle Ad-hoc Network (Vanet) di Area Kota Malang* adalah sebagai berikut :

1. Perlu dilakukannya penelitian lebih lanjut dengan menerapkan protokol lain routing lain.
2. Perlu dilakukan penambahan jumlah variasi *node* kendaraan dan kecepatan *node* untuk mendapatkan data yang lebih valid.



DAFTAR REFERENSI

- Abbasi, I., Khan, A., 2018. *A Review of vehicle to vehicle Communication Protocol for VANETs in the Urban Environment*. Malaysia: UNIMAS.
- Almohammed, A., Noordin, K., Saeed, S., 2016. *Evaluating the Impact of Transmission Range on the performance of the VANET*. Malaysia.
- Baskar, R., Chawla, M., 2015. *Vehicular Ad hoc Networks and its Application in Diversified Fields*. Maulana Azad Nation Institue of Technology.
- Bastian, R., Anggoro, R., Husni, M., 2017. *Implementasi Source Route pada Protokol GPSR dengan Bantuan Intersection Node untuk Meningkatkan Reliabilitas Pengiriman Data di VANET*. Indonesia. : Institut Teknologi Sepuluh Nopember.
- Bouras, C., Kapulas, V., Tsunai, E., 2015. *A GPSR Enhancement Mechanism for Routing in VANETs*. Greece : University of Patras.
- Deshmukh, A., Dorle, S., 2016. *Simulation of Urban Mobility (Sumo) for Evaluating Qos Parameter for Vehicular Adhoc Network*. India.
- Fogue, M., Sanguesa, J., Martinez, F., Barja, J., 2018. *Improving Roadside Unit Deployment in Vehicular Networks by Exploiting Genetic Algorithms*. Spain: University of Zaragoza.
- Goyal, P., Soni, A., 2014. *Simulation and Analysis of Routing Protocols in VANET using NS2 Simulator*. India: University Kurukshetra.
- Kafsi, M., P. Papadimitratos, O. Dousse, T. Alpcan, J.P. Hubaux. VANET connectivity alaysis. *IEEE Workshop on Automotive Networking and Applications(IEEE Computer Society)*, 2008. pp.1-10.
- Kumar, R., Dave, M., 2011. *A Comparative Study of Various Routing Protocols in VANET*. India.
- Paul, B., Ibrahim., Bikas., 2011. *VANET Routing Protocols: Pros and Cons*. Bangladesh: Shahjalal University.
- Prakasa, J., Anggoro, R., Wibison, W., 2016. *OPTIMASI KINERJA PROTOKOL AODV DENGAN STATIC INTERSECTION NODE*. Indonesia: ITS.
- Saputra, F., Anggoro, R., Djanali, S., 2018. *STUDI KINERJA GREEDY PERIMETER STATELESS ROUTING BERBASIS OVERLAY NETWORK PADA VANET*. Indonesia: Institut Teknologi Sepuluh Nopember University.
- Saini, M., Singh, M., 2015. *VANET, its Characteristics, Attacks and Routing Techniques : A Survey*. RC Gurdaspur: Guru Nanak Dev University.
- Saluja, A., Dargad, S., Mistry, K., 2017. *A Detailed Analogy of Network Simulators NS1, NS2, NS3, and NS4*. India : ITM University Vadorara.
- Xue, L., Yang, Y., Dong, D., 2016. *Roadside Infrastructure Planning Scheme for the Urban Vehicular Networks*. China.



LAMPIRAN

Lampiran 1 kode *throughput.awk*

```

1  # The BEGIN section, as noted above, is unnecessary.
2  BEGIN {
3      recvdSize=0
4      txsize=0
5      drpSize=0
6      startTime = 400
7      stopTime = 0
8      thru=0
9  }
10
11  {
12      event = $1
13      time = $2
14      node_id = $3
15      pkt_size = $8
16      level = $4
17
18      # Store start time
19      if (level == "AGT" && event == "s") {
20          if (time < startTime) {
21              startTime = time
22          }
23          # hdr_size = pkt_size % 400
24          # pkt_size -= hdr_size
25          # Store transmitted packet's size
26          txsize++;
27      }
28  }
29
30  # Update total received packets' size and store packets
31  arrival time
32
33  if (level == "AGT" && event == "r") {
34      if (time > stopTime) {
35          stopTime = time
36      }
37      # Rip off the header
38      # hdr_size = pkt_size % 400
39      # pkt_size -= hdr_size
40      # Store received packet's size
41      recvdSize++
42      # thru=(recvdSize/txsize)
43      # printf("%.2f %.2f \n", time, thru) > "tru2.tr"
44      if (level == "AGT" && event == "D") {
45          # hdr_size = pkt_size % 400
46          # pkt_size -= hdr_size
47          # Store received packet's size
48          drpSize++
49      }
50  }
51
52  END {

```




```
printf("Average
Throughput[kbps]=%.2f\n", (recvSize/(stopTime-
startTime)), txsize, drpSize, recvSize, startTime, stopTime);
}
```

Lampiran 2 kode *end-to-end delay.awk*

```
1 BEGIN {
2     seqno = -1;
3     # droppedPackets = 0;
4     # receivedPackets = 0;
5     count = 0;
6 }
7 {
8     if($4 == "AGT" && $1 == "s" && seqno < $6) {
9         seqno = $6;
10    }
11    if($4 == "AGT" && $1 == "s") {
12        start_time[$6] = $2;
13    } else if (($4 == "AGT") && ($7 == "cbr") && ($1 == "r")) {
14    }
15    end_time[$6] = $2;
16    } else if ($1 == "s" && $7 == "cbr") {
17        end_time[$6] = -1;
18    }
19 }
20 END {
21     for(i=0; i<=seqno; i++) {
22         if(end_time[i]>0) {
23             delay[i] = end_time[i] - start_time[i];
24             count++;
25         }
26         else{
27             delay[i] = -1;
28         }
29     }
30     for(i=0; i<=seqno; i++) {
31         if(delay[i]>0) {
32             n_to_n_delay = n_to_n_delay + delay[i];
33         }
34     }
35     n_to_n_delayAVG = n_to_n_delay/count;
36     print n_to_n_delayAVG * 1000 ;}
```




Lampiran 3 kode *packet delivery ratio.awk*

```

1 BEGIN {
2     recvdSize = 0
3     txsize=0
4     drpSize=0
5     startTime = 400
6     stopTime = 0
7     thru=0
8 }
9 {
10     event = $1
11     time = $2
12     node_id = $3
13     pkt_size = $8
14     level = $4
15
16     if (level == "RTR" && event == "r" && $7 == "cbr") {
17         receive++
18         if (time > stopTime) {
19             stopTime = time
20         }
21         recvdSize += pkt_size
22     }
23
24     if (level == "AGT" && event == "s" && $7 == "cbr") {
25         sent++
26         if (!startTime || (time < startTime)) {
27             startTime = time
28         }
29     }
30 }
31 }
32
33 END {
34     printf("PDR %.2f\n", (receive/sent)*100);
35 }

```