LAIR SÉRVULO DA CUNHA JÚNIOR

REG3M: REENCAMINHAMENTO GEOGRÁFICO DE MENSAGENS MULTIMÍDIA HOMOGRÁFICAS EM CONGESTIONAMENTOS VEICULARES

LAVRAS – MG

2020

LAIR SÉRVULO DA CUNHA JÚNIOR

REG3M: REENCAMINHAMENTO GEOGRÁFICO DE MENSAGENS MULTIMÍDIA HOMOGRÁFICAS EM CONGESTIONAMENTOS VEICULARES

Dissertação apresentado à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, área de concentração em Redes de Computadores e Sistemas Embarcados, para a obtenção do título de Mestre.

Prof. Dr. Luiz Henrique Andrade Correia
Orientador

LAVRAS – MG

2020

Ficha catalográfica elaborada pelo Sistema de Geração de Ficha Catalográfica da Biblioteca Universitária da UFLA, com dados informados pelo(a) próprio(a) autor(a)

Cunha Júnior, Lair Sérvulo da
ReG3M: Reencaminhamento Geográfico de Mensagens MultiMídia homográficas em congestionamentos veiculares / Lair Sérvulo da Cunha Júnior. - 2020.
86 p. :

Orientador: Prof. Dr. Luiz Henrique Andrade Correia.

Dissertação (Mestrado Acadêmico)–Universidade Federal de Lavras, 2020.
Bibliografia.

1. Redes veiculares. 2. Disseminação de mensagem. 3. Broadcast Storm. I. Correia, Luiz Henrique Andrade. II. Título.

LAIR SÉRVULO DA CUNHA JÚNIOR

REG3M: REENCAMINHAMENTO GEOGRÁFICO DE MENSAGENS MULTIMÍDIA HOMOGRÁFICAS EM CONGESTIONAMENTOS VEICULARES

Dissertação apresentado à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, área de concentração em Redes de Computadores e Sistemas Embarcados, para a obtenção do título de Mestre.

APROVADA em 21 de Agosto de 2020.

Prof. DSc. Raphael Winckler de Bettiol	UFLA
Prof. DSc. Demostenes Zegarra Rodriguez	UFLA
Profa. DSc. Daniel Ludovico Guidon	UFSJ

Prof. Dr. Luiz Henrique Andrade Correia
Orientador

LAVRAS – MG
2020

Aos meus pais, Lair e Ilza, e a toda a minha família que sempre me apoiaram e sem eles nada disso seria possível.

AGRADECIMENTOS

Agradeço primeiramente a minha família por estar sempre comigo.

Agradeço a Stefânia que sempre esteve ao meu lado e ajudou nos momentos difíceis.

Agradeço ao professor Luiz Correia, por todos os ensinamentos e orientações dadas ao longo do projeto.

Agradeço a todos os meus colegas de mestrado, por toda ajuda que recebi durante esta caminhada.

Agradeço a Universidade Federal de Lavras, especialmente ao Programa de Pós-Graduação em Ciência da Computação, pela oportunidade.

Muito Obrigado!

"And I won't look back, I can go the distance/And I'll stay on track, no I won't accept defeat"
(That's Phil's Boy)

RESUMO

O uso de tecnologia vem se tornando cada vez mais popular, com essa popularização, diversos setores passaram a usar as tecnologias, como redes veiculares. Redes veiculares ou VANETs, são redes formadas pela comunicação entre os próprios veículos (V2V), entre veículos e infraestrutura (V2I) e híbrida (V2X). Dentro de VANETs existem diversos tipos de aplicações, uma delas busca melhorar a eficiência no trânsito. Uma forma de melhorar a eficiência no trânsito é a utilização de vídeos para o monitoramento da vias. Em situações de congestionamento, os vídeos podem ser úteis para informar o estado em que se encontram os envolvidos no incidente que ocasionou o congestionamento. Uma maneira de enviar o vídeo para o maior número de pessoas possível é o envio por *broadcast*, porém esse tipo de envio pode ocasionar problemas. Para evitar problemas causados por envio em *broadcast*, este trabalho propõe um método de Reencaminhamento Geográfico de Mensagens MultiMídia (`ReG3M`). O ReG3M utiliza o veículo mais distante da origem da mensagem para ser o reencaminhador, também faz controle de mensagens repetidas na rede. O ReG3M foi comparado à transmissão em *broadcast* e aos algoritmos AID e DBRS. Este trabalho também propõe um *Framework* para utilização de homografia de vídeos em redes veiculares (VANET) em cenários de congestionamento. Os algoritmos SIFT, SURF, ORB e BRISK foram avaliados e comparados ao usarem o algoritmo RANSAC para a criação da matriz homográfica, considerando os algoritmos de qualidade de vídeo BRISQUE e NIQE. Resultados de simulação mostraram que o BRISK obteve melhor qualidade de imagem em relação aos outros descritores sendo também o segundo mais rápido em tempo de processamento. O `ReG3M` foi mais eficiente em quase todos os testes, no envio de mensagens do que os outros algoritmos, alcançando mais veículos e uma maior distância em menor tempo. Sendo melhor principalmente em mensagens com tamanhos maiores que seria o caso dos vídeos criados pelo processo de homografia.

Palavras-chave: *broadcast Storm*, transmissão, VANET, disseminação de mensagens, congestionamento, vídeo, homografia, descritores, qualidade de imagem

ABSTRACT

The use of technology has become increasingly popular, with this popularization, several sectors started to use as technologies, such as vehicular networks. Vehicle networks or VANETs, are networks formed by the communication between own vehicles (V2V), between vehicles and infrastructure (V2I) and hybrid (V2X). Within VANETs there are several types of applications, one of which seeks to improve traffic efficiency. One way to improve traffic efficiency is to use videos to monitor roads. In situations of congestion, videos can be useful to inform the state in which if the incident that caused the congestion occurs. One way to send the video to as many people as possible is to send by broadcast, however this type of sending can cause problems. To avoid problems caused by sending in broadcast, this work proposes a method of Geographical Forwarding of MultiMedia Messages (ReG3M). ReG3M uses the vehicle farthest from the message source to be the forwarder, it also controls repeated messages on the network. ReG3M was compared to transmission in broadcast and the AID and DBRS algorithms. This work also offers a Framework for using video homography in vehicular networks (VANET) in congestion scenarios. The SIFT, SURF, ORB and BRISK algorithms were obtained and compared when using the RANSAC algorithm to create the homographic matrix, considering the video quality algorithms BRISQUE and NIQE. Simulation results that BRISK obtained better image quality in relation to the other descriptors being also the second fastest in processing time. ReG3M was more efficient in almost all tests, in sending messages than the other algorithms, reaching more vehicles and a greater distance in less time. Being better mainly in messages with larger series sizes or in case of videos created by the homography process

Keywords: Broadcast Storm, transmission, VANET, message dissemination, congestion, video, homography, descriptors, image quality.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O uso de tecnologias vem se tornando cada vez mais popular, com isso vem surgindo aplicações tecnológicas no setor automobilístico, como as VANETs (*Vehicular Ad hoc Networks*), ou redes veiculares. A comunicação na *VANET* pode ser realizada entre os próprios veículos (V2V) usando dispositivos veiculares OBU (*On Board Unit*), entre veículos e a infraestrutura (V2I), através da RSU (*Road Side Unit*) e por comunicação híbrida (V2X) (PATEL; KAUSHIK, 2018). Em redes veiculares podem existir diversos tipos de aplicações, que basicamente são divididas em: aplicações de segurança no trânsito, como alertas de acidentes ou situações que podem gerar riscos aos condutores; aplicações de eficiência de transporte, que monitoram situações de trânsito; e aplicações de entretenimento, como acesso à Internet (HARTENSTEIN; LABERTEAUX, 2008). Dentro das aplicações de eficiência de trânsito, transmissões de vídeos relacionadas ao tráfego podem ser usadas para monitorar o trânsito e relatar situações de acidentes que possam ter ocorrido.

As situações de congestionamento podem surgir de diversas formas como acidentes, reformas na pista ou alguma situação de obstrução da via. Segundo o site (NEWS, 2018) o Brasil perde em média 267 bilhões de reais por ano devido aos congestionamentos. O jornal (TIMES, 2019) informa alguns malefícios decorrentes de congestionamentos, como por exemplo, a poluição gerada pelos escapamentos dos automóveis, que interfere no meio ambiente e que pode causar problemas respiratórios nas pessoas. Além disso, atrapalham também o bem estar psicológico, proveniente da sensação de desamparo que é sentida no trânsito e de sua imprevisibilidade. Os problemas causados pelo trânsito extremo também estão ligados ao aumento da violência doméstica, os autores (BELAND; BRENT, 2018) afirmam que nesses casos houve um aumento de 9% no número de incidentes.

Em um cenário de congestionamento, no qual todos os veículos são equipados com câmeras, é possível o compartilhamento de imagens pelo veículo que esteja posicionado mais próximo do incidente para todos os veículos que estejam neste congestionamento. A VANET pode ser utilizada para a transmissão e compartilhamento de vídeos curtos, com o intuito de informar e mostrar aos motoristas presentes no congestionamento, o que está ocorrendo à frente e o que fez com que eles estivessem nessa situação. Assim, o motorista pode tomar a decisão se deve continuar na via ou procurar uma rota alternativa. Podemos usar como exemplo, a situação de um acidente entre dois veículos, na qual os veículos envolvidos já estão sendo retirados da

via. O condutor ao receber um vídeo com esta informação, fica ciente que o congestionamento está quase no fim e que, em breve, o fluxo de veículos deve retornar ao normal.

Em uma situação deste tipo, os veículos ficam enfileirados e a troca de mensagens pode gerar um alto índice de colisões, gerando um problema conhecido como *broadcast storm*.

Este trabalho propõe um método de Reencaminhamento Geográfico de Mensagens MultiMídia (`ReG3M`). No ReG3M a escolha do veículo reencaminhador é baseada nas distâncias geográficas dos veículos em relação ao acidente, de forma a reduzir o número de reenvios e de colisões na rede e, com isso possibilitar o envio de mensagens *MultiMídia* como o envio de vídeos informativos dos incidentes que ocasionam os congestionamentos.

As imagens do incidente devem fornecer um campo de visão amplo e com uma qualidade de vídeo aceitável. Dessa forma, as imagens capturadas por dois ou mais veículos sobre o incidente devem ser reunidas para a melhorar a qualidade da imagem a ser compartilhada. Este processo é conhecido como homografia, que é o uso dos pontos característicos correspondentes em duas imagens que são unidos para se tornar apenas uma imagem. Com a homografia, é possível a criação de um mosaico de imagens, que consiste em um alinhamento de múltiplas imagens para a criação de apenas uma, aumentando o campo de visão de uma câmera sem perder a qualidade (GHOSH; KAABOUCH, 2016). Além disso, o tamanho da imagem transmitida deve ser reduzido para mitigar a latência de sua transmissão na VANET.

Para o processo de Homografia, é necessário utilizar algoritmos de descrição e detecção dos *keypoints* das imagens, como o SIFT (*Scale Invariant Feature Transform*), o SURF (*Speeded-Up Robust Features*), o ORB *(Oriented FAST and Rotated BRIEF)* e o BRISK *(Binary Robust Invariant Scalable Keypoints)* (LOWE, 2004; BAY et al., 2008; E.RUBLEE et al., 2011; LEUTENEGGER; CHLI; SIEGWART, 2011). E para reduzir o tamanho e reprojetar o ângulo das imagens, é empregado o algoritmo RANSAC (*Random Sample Consensus*) (FISCHLER; BOLLES, 1981). Este trabalho propõem também um *framework* para utilização de homografia em VANET, mostrando o processo de retirada e junção dos *frames* dos vídeos, comparando os principais descritores e analisando a qualidade dos vídeos gerados por esse processo.

1.1 Motivação

Em redes VANETs existem algumas aplicações que tem como objetivo melhorar a eficiência no trânsito. É importante para este tipo de aplicação alcançar o maior número de veículos. Para transmissões que buscam alcançar o maior número de pessoas possível, geralmente é uti-

lizado a transmissão em *broadcast*. Este tipo de transmissão pode ocasionar alguns problemas e os algoritmos que corrigem esse problema não são eficientes para mensagens grandes como mensagens de vídeo. Existem algoritmos que foram propostos para solucionar esse problema em VANET, porém esses algoritmos não conseguem ser eficientes para envios de mensagens maiores, como mensagens de vídeo.

1.2 Definição do problema

Alguns problemas que podem ocorrer em VANET, na transmissão das mensagens de vídeo utlizando *broadcast*, estão relacionados a *broadcast storm*. Com isso é necessário criar uma maneira de transmitir mensagens maiores como, vídeos, imagens ou áudios de modo à evitar os problemas causados pelo *broadcast*, como colisões na rede e o envio de mensagens repetidas.

1.3 Solução proposta

Para resolver o problema da transmissão de mensagens grandes, este trabalho apresenta o ReG3M (Reencaminhador Geográfico de Mensagens Multimídia), um protocolo de disseminação de mensagem com controle de *broadcast* baseado na distância do veículo receptor e do veículo de origem da mensagem. O ReG3M também controla as mensagens para que não exista recebimento de mensagens duplicadas.

Uma maneira de viabilizar o uso de vídeo para aplicação de eficiência no trânsito é criar a mensagem com um vídeo de curta duração, que será transmitida pelo ReG3M. A mensagem utiliza o vídeo gerado em uma câmera que esteja nos veículos mais próximos da situação que está causando o bloqueio do trânsito. A partir dos veículos, que estão próximos ao incidente, o vídeo seria transmitido para veículos que vieram atrás no congestionamento. Para melhorar o campo de visão do ocorrido e melhorar a qualidade do vídeo, pode ser usado imagens das câmeras de dois ou mais veículos. Os vídeos devem passar por um processo de homografia, que irá uni-los para se tornar apenas um vídeo.

1.4 Objetivo Geral

Este trabalho tem como proposta, transmissão de vídeo como aplicação de eficiência no trânsito. Para isso é necessário alcançar dois objetivos, o primeiro objetivo é realizar a trans-

missão das mensagens que tenham os vídeos curtos, corrigindo problemas de *broadcast storm* no envio de mensagens *Multimídia*. Foi criado um protocolo de disseminação de mensagens, ReG3M que controla o reenvio de mensagens e evita o recebimento de mensagens duplicadas. O ReG3M também pode ser utilizado para envio de dados além dos vídeos curtos como, áudios e imagens. O segundo é utilizar o processo de homografia para unir vídeos de dois ou mais veículos mantendo a qualidade dos vídeos e aumentando o ângulo de visão. O vídeo será transmitido de veículos que tenham melhor visão do que está ocorrendo à sua frente, os vídeos serão unidos e transmitidos para os veículos que estiverem atrás e, com obstrução visual.

1.5 Objetivos específicos

Os objetivos específicos deste trabalho para a transmissão de imagens em redes veiculares, consistem em:

i) desenvolvimento de um método de Reencaminhamento Geográfico de Mensagens MultiMídia (`ReG3M`) para redução do número de retransmissões e colisões na rede;

ii) simulação de uma VANET com transmissão e retransmissão de vídeo, em situação de congestionamento e alta densidade de veículos;

iii) Comparar protocolos que buscam corrigir os problemas de *broadcast storm*;

iv) implementação de um *Framework* para utilização de homografia de vídeos em VANETs, para a transmissão de vídeos informativos em situações de acidente com maior ângulo de visão;

v) comparar algoritmos de descrição e detecção de pontos característicos de imagens.

1.6 Organização do texto

Este trabalho está organizado da seguinte forma: O Capítulo 2 apresenta os conceitos utilizados como base para o trabalho. O Capítulo 3 apresenta trabalhos relacionados sobre o uso de vídeo em VANET e protocolos que buscam corrigir problemas de *broadcast storm*. O capítulo 4 apresenta o ReG3M (Reencaminhador Geográfico de Mensagens Multimídia). O Capítulo 5 apresenta a metodologia utilizada para o desenvolvimento do *framework* para o uso de homografia em VANET. O Capítulo 6 apresenta os resultados obtidos após desenvolvimento

e testes da solução proposta. Por fim, o Capítulo 7, apresenta a conclusão dos trabalhos e os trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo é apresentado o referencial teórico que enfoca os conceitos utilizados como base para o trabalho. O capítulo descreve alguns conceitos de redes ad hoc e MANET, abordando o que é uma rede veicular (VANET) e seus componentes. Outros conceitos a respeito da arquitetura WAVE para VANET, também são apresentados. Além disso são apresentadas aplicações em VANET e disseminação de mensagens por *Broadcast*. São introduzidos conceitos de visão computacional, algoritmos de detecção e descrição de imagens, homografia e métricas de qualidade de imagem.

2.1 Redes Ad Hoc

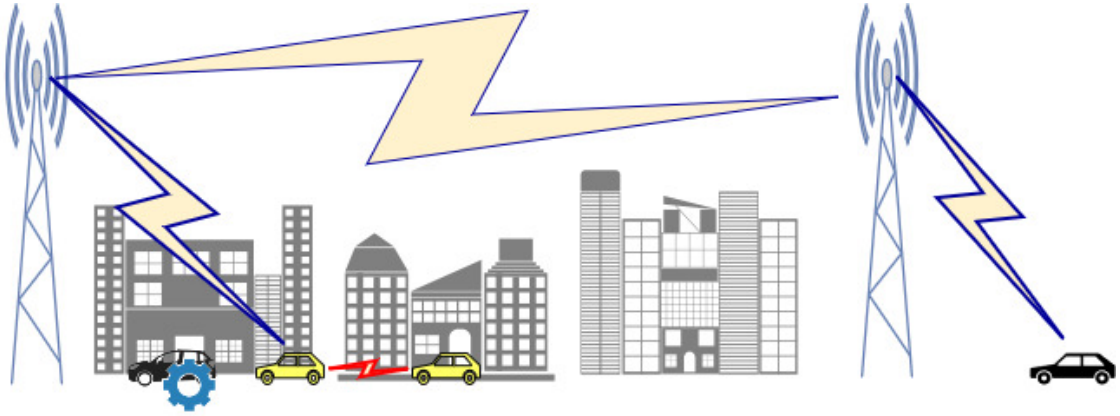
Uma rede *ad hoc* possui *hosts* móveis sem fio que formam uma rede temporária sem o auxílio de qualquer infraestrutura estabelecida ou administração centralizada. Em tal ambiente, pode ser necessário que um *host* móvel solicite o auxílio de outros *hosts* para encaminhar um pacote ao seu destino, devido ao alcance limitado das transmissões sem fio de cada *host* móvel (JOHNSON; MALTZ, 1996)

As *Mobile Ad Hoc Network* (MANET) são um tipo de redes Ad Hoc com características especiais, como topologia dinâmica, rede distribuída, roteamento multihop, heterogeneidade de dispositivos e restrição de energia (HOEBEKE et al., 2004). Entre as MANET existe a *Rede Ad hoc Veicular* (VANET) que é utilizada para fornecer comunicações entre veículos nas proximidades e, entre veículos e infraestrutura ao longo da estrada (RANJAN; AHIRWAR, 2011).

2.1.1 *Vehicular Ad hoc Networks (VANET)*

Com o aumento do uso da tecnologia em áreas diversas sugiram tipos de redes de computadores, uma dessas redes é a *Vehicular Ad hoc Networks* (VANET). Essa rede é formada através da comunicação entre veículos e dispositivos que ficam em rodovias. Essa comunicação irá permitir um novo conjunto de aplicações para segurança e o conforto de condução. Por exemplo, um veículo pode avisar outros veículos sobre o tráfego, acidentes ou condições de estradas ruins (HARSCH; FESTAG; PAPADIMITRATOS, 2007). A Figura 2.1 ilustra uma rede veicular enviando uma mensagem de alerta para avisar que um veículo está parado na via.

Figura 2.1 – Rede veicular



Fonte: Do Autor

As redes veiculares possuem algumas características próprias se comparadas com alguma outra MANET, entre elas estão:

1. Mobilidade previsível dos nós: Os nós de uma VANET tem um trajeto definido para percorrer, tendo que respeitar sinalizações de trânsito e estradas que formam seu trajeto (LI; WANG, 2007);

2. Não possui limitações de energia: Os veículos conseguem fornecer energia constante para as OBUs (JAKUBIAK; KOUCHERYAVY, 2008);

3. Densidade da rede: A densidade da rede varia de acordo com as condições do trânsito, pode ser muito alta em casos de congestionamento ou muito baixa em casos aonde o fluxo de veículos é menor (TOOR et al., 2008);

4. Mudança de topologia: A topologia em VANET é dinâmica devido à movimentação e velocidade dos veículos (JAKUBIAK; KOUCHERYAVY, 2008);

5. Alta capacidade computacional: Os veículos podem ser equipados com sensores e recursos computacionais robustos como processadores, grande capacidade de memória, tecnologia avançada de antenas e sistema de posicionamento global (GPS) (OLARIU; WEIGLE, 2009).

2.1.2 Comunicação em VANET

Para existir comunicação em VANET os veículos participantes são equipados com dispositivos de comunicação *(On Board Unit ou OBU)* que são responsáveis pelas trocas de infor-

mações com outros veículos que possuem uma OBU ou com as unidades ao longo da estrada (*Road side units* ou RSUs).

A OBU possui uma interface especializada para se conectar com outras OBUs e um dispositivo de rede sem fio baseado no documento IEEE 802.11p. A OBU faz as funções do veículo como emissor, receptor e roteador de pacotes (AL-SULTAN et al., 2014).

Outro dispositivo de uma VANET é a RSU, ele fica fixado ao longo das rodovias em lugares estratégicos. Assim como a OBU possui um dispositivo de rede sem fio de baseado no IEEE 802.11p, as RSUs comunicam com veículos ou outras RSUs. Uma das funções da RSU é aumentar o alcance da comunicação transmitindo informações para outras RSUs que transmitem para outras OBUs. Essas informações podem ser alertas de segurança ou informações do trânsito em geral (AL-SULTAN et al., 2014).

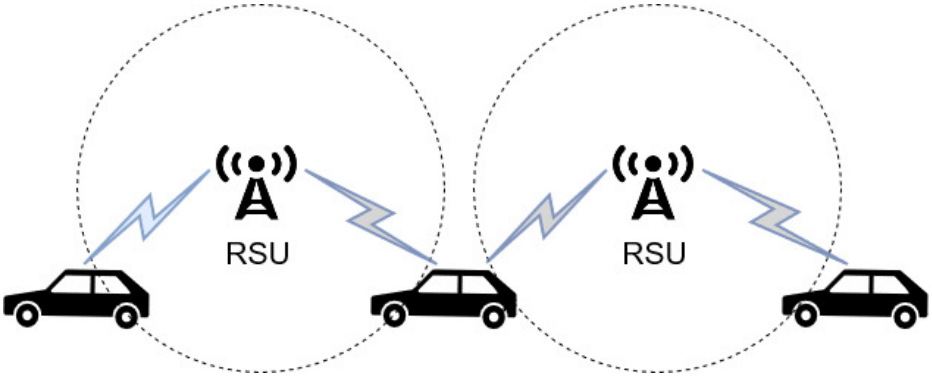
2.2 *Wireless access in vehicular environments* (WAVE)

O WAVE é uma arquitetura de comunicação de rádio que fornece serviços para comunicação entre veículo para veículo (V2V) mostrada na Figura 2.2, veículo para a infraestrutura (V2I) mostrada na Figura 2.3 e comunicação híbrida (V2X) mostrada na Figura 2.4, no sistema de transporte inteligente ou *Intelligent Transportation System* (ITS) (PATEL; KAUSHIK, 2018).

Figura 2.2 – Comunicação V2V

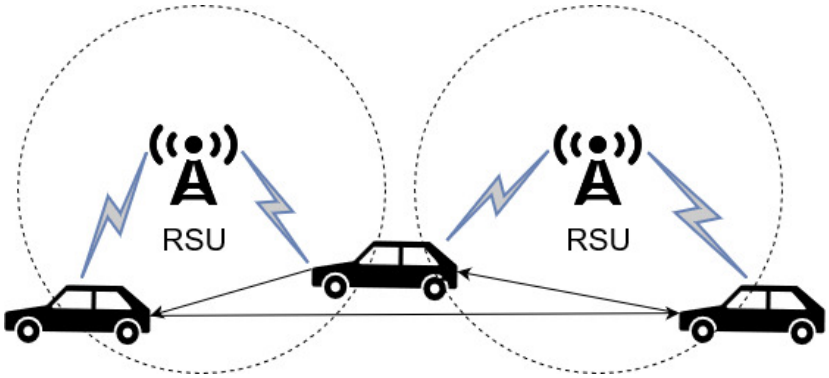
Fonte: Do autor (2019)

Figura 2.3 – Comunicação V2I



Fonte: Do autor (2019)

Figura 2.4 – Comunicação V2X



Fonte: Do autor (2019)

Em 2004 o grupo IEEE 802.11 começou a desenvolver um padrão de comunicação para as redes veiculares denominado de IEEE 802.11p. Outro grupo da IEEE assumiu a tarefa de incluir camadas adicionais no conjunto de protocolos, formando o IEEE 1609. Em conjunto, o IEEE 802.11p e o IEEE 1609.x são chamados de *Wireless Access in Vehicular Environments* (WAVE) (UZCATEGUI; SUCRE; ACOSTA-MARUM, 2009). O grupo IEEE 802.11p publicou um conjunto de especificações da camada física e de controle de acesso ao meio (MAC) para comunicação em ambientes dinâmicos na faixa de frequência *Dedicated Short Range Communication* (DSRC) de 5,85 a 5,925 GHz. (CAMPOLO; MOLINARO, 2011). O WAVE é composto por alguns documentos, IEEE 802.11 e IEEE 802.11p e os documentos do padrão IEEE 1609 (JIANG; DELGROSSI, 2008)

A família *WAVE* IEEE 1609 define um conjunto complementar de protocolos, serviços e padronização em interfaces *WAVE* para comunicação em VANET. A família de padrões*WAVE* IEEE 1609 é organizada da seguinte forma (IEEE, 2009):

1. *IEEE P1609.0 Draft Standard for Wireless Access in Vehicular Environments (WAVE)*: Descreve os serviços necessários para dispositivos DSRC/WAVE multicanais se comunicarem em um ambiente veicular móvel;

2. *IEEE 1609.1-2006 - Trial Use Standard for Wireless Access in Vehicular Environments (WAVE)*: Define o formato das mensagens de comando e as resposta para essas mensagens, formato de dados que devem ser armazenados para a comunicação entre os aplicativos e componentes da rede;

3. IEEE 1609.2 -2006: Define o formato para mensagens de segurança. Esse padrão também define as circunstâncias para o uso de trocas de mensagens;

4. IEEE 1609.3 -2007: Define serviços de camada de rede e de transporte, incluindo endereçamento e roteamento, em suporte à troca segura de dados;

5. IEEE 1609.4 -2006: Fornece aprimoramentos ao Controle de Acesso ao Meio (MAC) IEEE 802.11 para suportar operações *WAVE*.

O WAVE serve como um dos padrões de comunicação em VANET e, destina-se a funções e serviços requeridos por estações que seguem esse padrão para trocas rápidas de mensagens (JIANG; DELGROSSI, 2008).

2.3 Aplicações VANET

Existem diversos tipos de aplicações e projetos em VANET, segundo *IEEE (2009*) as aplicações podem ser divididas em três tipos:

1. Aplicações de segurança no trânsito, como alertas de acidentes ou situações que podem gerar riscos aos condutores, mensagens desse tipo de aplicação têm prioridade;

2. Aplicações de eficiência de transporte, que monitoram situações de trânsito como congestionamento, alagamentos ou obras;

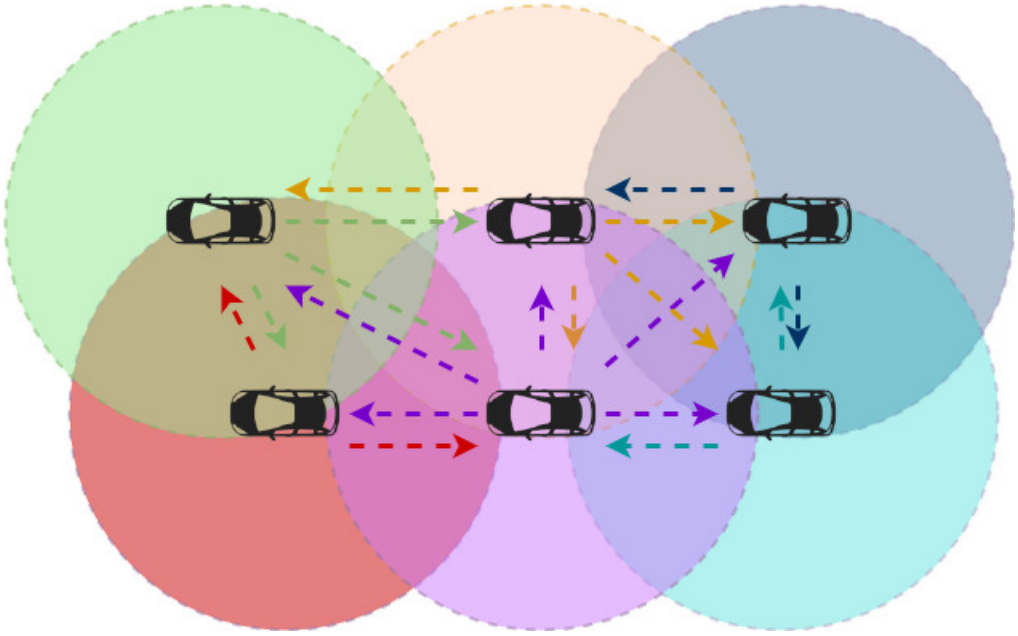
3. Aplicações de entretenimento, como acesso à internet, transmissão de vídeos e músicas.

As aplicações em VANET são possíveis com o uso de equipamentos utilizados para fazer a comunicação, podendo estar nos veículos ou nas rodovias.

2.4 Disseminação de mensagens por *Broadcast*

Existem algumas maneira para a disseminação de mensagem em VANET, entre elas está a disseminação por *Broadcast*. Tradicionalmente uma das formas de disseminação em *broadcast* é o *flooding*. Esta forma de disseminação funciona com o envio da mensagem do nó de origem para todos os seus vizinhos, os nós vizinhos que receberam a mensagem pela primeira vez tem a obrigação de reenviar a mensagem para todos os seus vizinhos, isso ocorre até que toda a rede receba a mensagem (P.NAND; SHARMA, 2011). O uso do *flooding* pode causar problemas na rede, devido a quantidade de retransmissões, colisões e mensagens redundantes, este problema é chamado de *broadcast Storm* (TSENG; NI; SHIH, 2003). A Figura 2.5 mostra uma representação de envio de mensagem por *broadcast*, onde os veículos transmitem a mensagem para todos que estiverem em seu alcance.

Figura 2.5 – Transmissão *broadcast*



Fonte: Do Autor

Os problemas que podem ser causados pela *broadcast storm* são (TSENG; NI; SHIH, 2003):

Redundância: Reenvio de mensagens aonde todos os nós vizinhos já tenham recebido a mensagem;

Contenção: Muitas mensagens podem ser recebidas pelos mesmos veículos, isso faz com que ele demore para fazer o seu envio e o que faz com que as mensagens fiquem contidas;

Colisão: Envios simultâneos de mensagens pelos veículos, pode ocasionar colisão na rede, o que impede que transmissão seja realizada.

Os protocolos baseados em *broadcast* podem ser classificados de duas formas, de acordo como é feito o *flooding* na rede. Podendo ser um *flooding controlado* ou *flooding não controlado* (KUROSE; ROSS, 2009):.

***Flooding não controlado*:** É quando todos os nós vizinhos que receberam a mensagem reenviam a mensagens para todos os seus vizinhos;

***Flooding controlado*:** É quando existem alguns mecanismos para diminuir a quantidade de mensagens enviadas. Esses mecanismos podem ser, comparação de mensagens para evitar redundância, escolha de um nó que ficará responsável pelo envio, entre outros.

Existem alguns algoritmos que podem diminuir ou evitar problemas em envios em *broadcast*. Algumas das formas de controlar o *flooding* e tentar evitar o *Broadcast Storm*, foram citados por (SILVA; MINI; CUNHA, 2017):

- **Baseado em atraso**: Cada veículo tem um tempo aleatório para fazer o reenvio das mensagens para evitar transmissões ao mesmo tempo;
- **Baseado em *Cluster***: é escolhida um veículo com *Cluster* que será responsável pelo reenvio das mensagens em uma determinada área;
- **Baseado em Contador**: É analisado a quantidade de mensagens repetidas e com base nessa análise o veículo decide se faz ou não o reenvio;
- **Baseado em distância**: Os veículos mais distantes da origem da mensagem tem prioridade para o reenvio.

2.5 Compartilhamento de vídeo em VANET

O compartilhamento de vídeos é um tipo de transmissão que necessita de grande quantidade de recursos para ser executada. Contudo, os avanços no desenvolvimento de *hardware* trouxe essa capacidade para dispositivos menores e heterogêneos como celulares e câmeras de vídeo pequenas e também para centrais multimídias ou OBUs instaladas em veículos. Essa evolução do *hardware* fez com que o compartilhamento de vídeo possa ser usado nas redes móveis ad hoc (LINDEBERG et al., 2011).

Melhorias em VANET como melhores equipamentos de *hardware* empregado nos veículos ou RSU, e o aumento da velocidade de transmissão na rede, possibilitam a captura e o compartilhamento de serviços multimídias. Dessa forma, os veículos podem transmitir informações em vídeos e ajudar no monitoramento das vias rodoviárias (FELICE et al., 2015).

2.6 Visão Computacional

A visão humana pode perceber estruturas tridimensionais, podemos distinguir formas, reconhecer objetos, pessoas e, em alguns casos, até mesmo emoções, somente com um olhar. Pesquisadores de visão computacional tentam reproduzir essas percepções em computadores (SZELISKI, 2010). Contudo a Visão computacional é a transformação de um dado capturado por uma câmera em uma decisão ou uma nova representação. Diferente da visão humana o computador consegue ver apenas números nas imagens capturadas (BRADSKI; KAEHLER, 2008). Atualmente, a visão computacional vem sendo utilizada em diversas áreas, como (SZELISKI, 2010):

- ***Optical character recognition (OCR)*:** Leituras de manuscritos;
- **Inspeção de máquinas**: Inspecionando máquinas em busca de defeitos nas peças;
- **Construção de modelo 3D (fotogrametria):** construção totalmente automatizada de modelos 3D a partir de fotografias aéreas usadas em sistemas como o *Bing Maps*;
- **Segurança automotiva:** detecção de obstáculos inesperados;
- **Vigilância:** monitoramento de intrusos, análise do tráfego nas rodovias.

Além dessas podemos ver seu uso em reconhecimento facial, autentificação visual, reconstrução 3D de um cenário, realidade aumentada, realidade virtual, etc (SZELISKI, 2010)

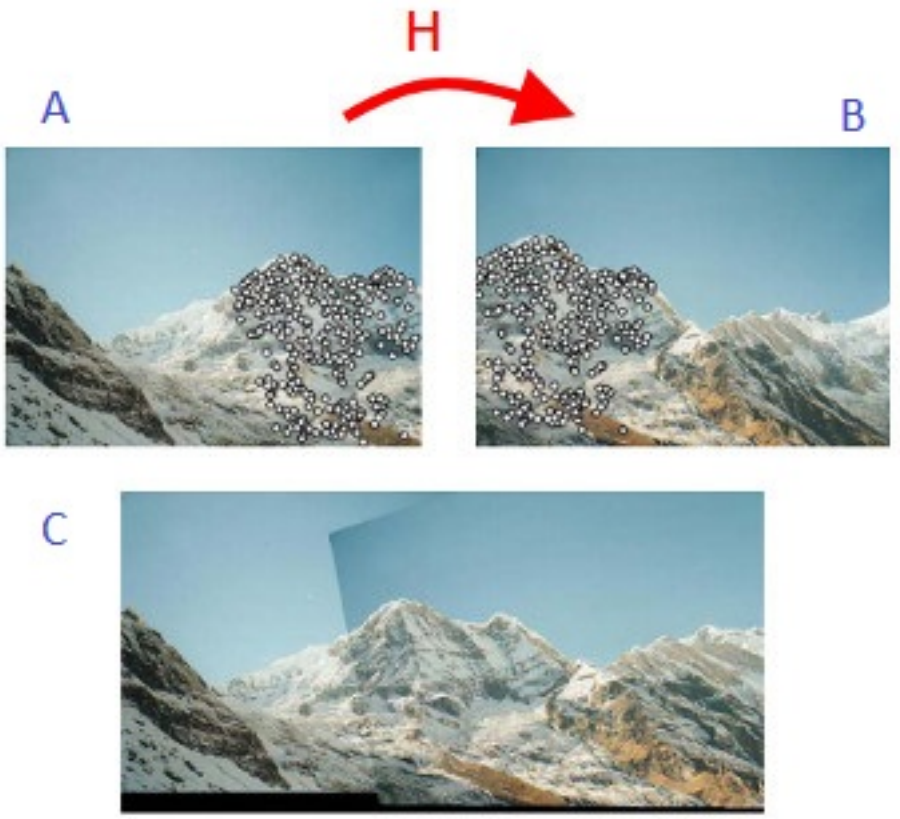
2.7 Homografia

A detecção e correspondência de características de uma imagem é essencial para muitas aplicações da visão computacional (SZELISKI, 2010). A homografia utiliza desse recurso para encontrar pontos semelhantes em duas imagens de um mesmo objeto. O BFMatch (*Brute Force Match*) busca pontos correspondentes em todos os pontos das imagens (VISION, 2018). Após encontrar é possível unir as duas imagens, mesmo que elas estejam em ângulos diferente, formando apenas uma nova imagem (LEE; YOON; LIM, 2017). Na Figura 2.6 podemos ver que *A* e *B* representam as imagens que irão ser unidas, nelas podemos ver também destacados, os pontos característicos de cada imagem. O resultado da união é mostrado em *C*.

Alguns dos passos da homografia são a reprojeção, que é o alinhamento dos *pixels* e o *overlay* que faz a sobreposição das imagens, mesclando valores de *pixels*. Alguns algoritmos de detecção e descrição são usados para a mescla das imagens na etapa de *overlay*. Com a homografia, é possível a criação de um mosaico de imagens, que consiste em um alinhamento de múltiplas imagens para a criação de apenas uma, aumentando o campo de visão de uma câmera sem perder a qualidade (GHOSH; KAABOUCH, 2016).

Para a união das imagens geradas é necessário a calibração das câmeras. A matriz homográfica representa os pontos semelhantes das imagens. A matriz homográfica mapeia uma transformação do plano projetivo real para o plano projetivo virtual, fazendo as correções geradas por essa transformação. A matriz homográfica possui 8 graus de liberdade no que diz respeito a: isometria, transformação de similaridade, transformação afim, transformação projetiva e projeção em perspectiva (BROWN; LOWE, 2006).

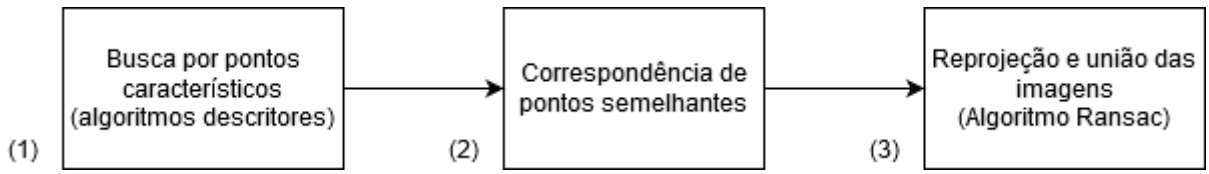
Figura 2.6 – *Homography Panorama stitching*



Fonte: <http://man.hubwiz.com/manual/OpenCV>

O processo para a junção de duas imagens por homografia é mostrada na figura 2.7. Em (1) é realizado a busca pelos pontos característicos das imagens, após isso em (2) é feita a correspondência entre pontos semelhantes das duas imagens, em seguida (3) é realizada a reprojeção e união das imagens.

Figura 2.7 – Processo de junção das imagens



Fonte: do Autor

2.7.1 Detecção e descrição dos recursos de imagem

Extrair características de imagem tem sido considerado um dos tópicos mais ativos para representação de imagem na comunidade de visão computacional. A extração de recursos envolve detectar e isolar os recursos desejados da imagem ou padrão, para identificar ou inter-

pretar informações significativas dos dados da imagem. As características podem ser divididas em características globais, quando apresentam o conteúdo de toda a imagem, ou locais quando apresentam o conteúdo de pequenos trechos da imagem (MIKOLAJCZYK et al., 2005). Como as características globais visam representar a imagem como um todo, apenas um único vetor de característica é produzido por imagem e, portanto, o conteúdo de duas imagens pode ser comparado através da comparação de seus vetores. Já nas locais a extração de características geralmente pode ser dividida em duas etapas independentes: detecção e descrição (MOREELS; PERONA, 2007). Existem muitos tipos de características de imagem que podem ser extraídas, como bordas, bolhas, cantos, pontos de interesse, textura e cor (MIKOLAJCZYK et al., 2005). Um grande número de algoritmos de extração de características foi proposto na literatura para fornecer uma correspondência confiável de características, como apresentados a seguir.

2.7.1.1 *Scale Invariant Feature Transform* (SIFT)

Segundo TAREEN; SALEEM (2018) o *Scale Invariant Feature Transform* (SIFT) é um dos mais renomados algoritmo de descrição e detecção de características. O SIFT transforma as coordenadas invariantes em escalas relativas às características locais da imagem. Com essa abordagem é possível gerar um número de pontos característicos que cobrem densamente a imagem. Por exemplo, uma imagem 500 x 500 *pixels* pode dar origem até 2.000 pontos dependendo do conteúdo da imagem. A quantidade de pontos é importante para o reconhecimento dos objetos, ao menos três pontos devem ser correspondidos corretamente para a identificação do objeto. Os pontos são encontrados através de uma pesquisa no máximo de locais possíveis da imagem, após encontrar os pontos cada ponto é dividido em sub-blocos (TAREEN; SALEEM, 2018).

Para reconhecimento de um objeto os pontos característicos são extraídos de um conjunto de imagens e, em seguida, são armazenados. Quando uma nova imagem é comparada, seus pontos característicos são comparados com pontos armazenados e, com a correspondência desses pontos é feita a identificação do objeto. Imagens desordenadas podem gerar pontos falsos que devem ser descartados (LOWE, 2004).

As etapas para se obter as características das imagens são (LOWE, 2004):

1. Detecção espacial-escalar: Faz uma busca em todos as escalas e locais da imagem. Utiliza a função diferença de Gauss (DoG) para identificar pontos de interesse que são invariantes à escala e orientação. A função *DoG* é formada pela diferença de imagens

filtradas em escalas próximas, separadas por uma constante de escala k dada por:

$$DoG = G(x,y,Ks) - (x,y,s) \tag{2.1}$$

2. Localização de pontos-chave: Tem a finalidade de eliminar os pontos com baixo contraste ou mal localizados. Usando a função DoG ocorre instabilidade nos pontos de interesse localizado nas bordas. A eliminação desses pontos é realizado com a matriz Hessiana (H) 2x2 da função DoG. A partir da matriz H é feita uma razão do traço $Tr(H)$ pelo determinante $det(H)$ em relação dos autovalores (maior/menor) de H. A formula é dada pela Equação 2.2:

$$\frac{Tr(H)^2}{Det(h)} < \frac{(r+1)^2}{r} \tag{2.2}$$

3. Atribuição de orientação: Para cada ponto-chave é adicionada orientações para construir descritores invariantes à rotação.

4. Descritor de ponto-chave: Os descritores são criados levando em consideração as características de cada ponto-chave. Para cada imagem é construído diversos descritores que serão utilizados para comparação de semelhanças entre duas imagens.

   Os gradientes locais da imagem são medidos na escala selecionada na região ao redor de cada ponto-chave. Estes são transformados em uma representação que permite níveis significantes de distorção de forma e mudanças na iluminação.

2.7.1.2 *Speeded-Up Robust Features* (SURF)

Proposto por BAY et al. (2008) o algoritmo SURF utiliza a determinante da Matriz Hessiana para a criação de um detector e um descritor baseado em distribuição. Explora imagens integrais para melhorar a velocidade de detecção de características.

A Equação 2.3 representa a matriz hessiana no ponto $x = (x,y)$ na escala de σ, onde $L_{xx}(X,\sigma)$ convolução gaussiana de segunda ordem derivada de $\frac{\delta^2}{\delta x^2} g(\sigma)$ de uma imagem no ponto x e da mesma forma para $L_{yy}(X,\sigma)$ e $L_{xy}(X,\sigma)$:

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(X,\sigma) & L_{xy}(X,\sigma) \\ L_{xy}(X,\sigma) & L_{yy}(X,\sigma) \end{bmatrix} \tag{2.3}$$

A principal vantagem do SURF sobre o SIFT é seu baixo custo computacional. Dentro das características do SURF estão (BAY et al., 2008):

1. Detecção de pontos chave: O algoritmo SURF utiliza uma aproximação da matriz Hessiana. Utiliza imagens integrais pois aumenta a resposta do algoritmo

2. Descrição dos pontos chave: O algoritmo busca a identificação da orientação dos pontos-chaves. Para a criação de um descritor é necessária construir uma região quadrada centrada ao redor do ponto de interesse e orientada ao longo da orientação selecionada para o ponto de interesse.

2.7.1.3 *Oriented FAST and rotated BRIEF*(ORB)

O ORB (*Oriented FAST and Rotated BRIEF*) proposto por (E.RUBLEE et al., 2011) é uma junção entre os algoritmos FAST *(Features from Accelerated Segment Test)* de (ROSTEN; DRUMMOND, 2006) e o BRIEF *(Binary Robust Independent Elementary Features)* de (CALONDER et al., 2010). Os pontos das imagens são localizados nos cantos de uma segmentação em forma de pirâmide e avaliados usando a pontuação Harris Corner para filtrar os pontos de alta qualidade. Os recursos do ORB são invariantes à escala, rotação e alterações afins limitadas.

2.7.1.4 *Binary Robust Invariant Scalable Keypoints* (BRISK)

O BRISK (*Binary Robust Invariant Scalable Keypoints*) proposto por (LEUTENEGGER; CHLI; SIEGWART, 2011) também utiliza o algoritmo FAST para busca dos pontos. A descrição do BRISK é baseada na identificação da direção característica de cada recurso para obter invariância de rotação. Para atender à invariância da iluminação, os testes simples de brilho também são concatenados e o descritor é construído como uma sequência binária. Os recursos do BRISK são invariantes à escala, rotação e alterações afins limitadas.

2.7.2 *Random sample Consensus* (RANSAC)

O *Random sample Consensus* (RANSAC) é um método de estimação para a extração de *Inliers* de um conjunto de características. Ele faz o cálculo da matriz homográfica H, utilizando o método direto de transformação linear (DLT) (BROWN; LOWE, 2007). A matriz homográfica mapeia uma transformação do plano projetivo real para o plano projetivo resultante, fazendo as correções geradas por essa transformação. A matriz homográfica possui 8 graus de liberdade no

que diz respeito a isometria, transformação de similaridade, transformação afim, transformação projetiva e projeção em perspectiva (BROWN; LOWE, 2006).

2.8 Qualidade de imagem

As imagens digitais estão sujeitas a uma ampla variedade de distorções durante a aquisição, processamento, armazenamento, transmissão e reprodução. Esses tipos de distorções podem ocasionar uma degradação da qualidade visual. Com isso, mediar a qualidade da imagem é muito importante para inúmeras aplicações de processamento de imagens (KAUR; JYOTI, 2011). A avaliação da qualidade pode ser realizada de forma subjetiva feita pela visão humana ou por métricas objetivas de qualidade de imagem, que se correlacionam com a qualidade percebida (KAUR; JYOTI, 2011).

O sistema visual humano (*Human Vision System* HVS) é a métrica mais confiável para avaliação da qualidade de imagem (ITU-T, 2008). No entanto, é um processo complexo e demorado de avaliação, o que o torna inaplicável no ambiente de processamento de imagens em tempo real (GVOZDEN; GRGIC; GRGIC, 2018).

Para a avaliação objetiva de uma imagem pode ser utilizado algoritmos que não precisam de uma imagem como referência para comparação, estes algoritmos são classificados como *No Reference (NR)* (AHMED; CHEN; HAMMAD, 2017):

2.8.1 *No Reference Quality Metrics*

Avaliação de qualidade de imagem *Blind* ou *No reference (NR)*, refere-se à avaliação automática de qualidade de uma imagem, usando um algoritmo. O algoritmo é capaz de avaliar a qualidade da imagem utilizando apenas a imagem que está sendo avaliada (MITTAL; MOORTHY; BOVIK, 2012). Entre os NR é possível citar os algorimtos BRISQUE (*Blind/Referenceless Image Spatial Quality Evaluator*) (MITTAL; MOORTHY; BOVIK, 2012) e o NIQE (*Naturalness Image Quality Evaluator*) (MITTAL; SOUNDARARAJAN; BOVIK, 2013). Ambos os algoritmos treinam um modelo usando características estatísticas previsíveis idênticas, chamadas de *natural scene statistics (NSS)*. O NSS são baseados em coeficientes de luminância normalizados no domínio espacial, e são modelados como uma distribuição gaussiana multidimensional. Distorções aparecem como perturbações na distribuição gaussiana (MITTAL; MOORTHY; BOVIK, 2012).

O BRISQUE limita-se a medir a qualidade das imagens com o mesmo tipo de distorção do modelo. Um modelo BRISQUE é treinado usando pontuações subjetivas de opinião, com a vantagem de que o escore BRISQUE se correlaciona bem com a percepção humana de qualidade (MITTAL; MOORTHY; BOVIK, 2012). O modelo, não calcula características específicas de distorção, como desfoque ou bloqueio, mas usa estatísticas de cena de coeficientes de luminância localmente normalizados para quantificar possíveis perdas de "naturalidade" na imagem, devido à presença de distorções, levando a uma medida holística da qualidade. As características subjacentes usadas derivam da distribuição empírica de luminâncias localmente normalizadas e produtos de luminâncias, localmente normalizadas sob um modelo estatístico de cena natural espacial (MITTAL; MOORTHY; BOVIK, 2012).

O BRISQUE tem uma baixa complexidade computacional, tornando-o adequado para aplicações em tempo real. As características do BRISQUE também podem ser usadas para identificação de distorção (MITTAL; MOORTHY; BOVIK, 2012).

O NIQE, baseia-se na construção de uma coleção de recursos estatísticos "com qualidade" baseada em uma cena natural de domínio espacial simples e bem-sucedida. É um modelo IQA (*Image Quality Assurance*) cego que utiliza apenas desvios mensuráveis de regularidades estatísticas observadas em imagens naturais, sem treinamento em imagens distorcidas com classificação humana e, de fato, sem qualquer exposição a imagens distorcidas. A qualidade da imagem distorcida é expressa como uma métrica de distância simples entre as estatísticas do modelo e as da imagem distorcida (MITTAL; SOUNDARARAJAN; BOVIK, 2013). O NIQE extrai os recursos do NSS de blocos estatisticamente significativos na imagem distorcida. A função se ajusta a uma distribuição Gaussiana multivariada para os recursos do NSS da imagem. O índice de qualidade é a distância entre as distribuições gaussianas (MITTAL; SOUNDARARAJAN; BOVIK, 2013).

Em comparação ao BRISQUE, o índice NIQE não está vinculado a nenhum tipo específico de distorção mas, fornece poder preditivo quase comparável nas mesmas distorções em que o índice BRISQUE foi treinado, com uma baixa complexidade (MITTAL; SOUNDARARAJAN; BOVIK, 2013).

3 TRABALHOS RELACIONADOS

Quanto a transmissão de mensagens, algumas soluções para mitigar o problema do *broadcast storm* são encontradas na literatura. O protocolo AID *(Approach for Information Dissemination)* (BAKHOUYA; GABER; LORENZ, 2011) visa o controle de mensagens enviadas em *broadcast* para a rede. Esse controle é baseado na densidade de mensagens, que decide reenviar ou não a mensagem, de acordo com a quantidade de mensagens duplicadas recebidas em um determinado intervalo de tempo. O algoritmo possui um contador aleatório τ, esse contador será o tempo de espera para os cálculos de densidade das mensagens. Após o tempo τ expirar os veículos irão calcular se a densidade de mensagens repetidas foi baixa e assim retransmitir a mensagem. O problema do AID é o particionamento de rede, quando o número de veículos na área de interesse não é suficiente para realizar retransmissões de mensagens.

Outro algoritmo para controle de *broadcast* é o DBRS *(Distance Based Relay Selection)* proposto por (KIM et al., 2008). A distância entre os veículos de origem e destino é usada para decidir se o veículo que recebeu a mensagem irá retransmitir. Para selecionar o retransmissor cada veículo terá um tempo de espera antes de fazer o reenvio, esse tempo é inversamente proporcional à distância ($Tempo = \frac{1}{distancia}$). Dessa forma, o veículo mais distante faz o reenvio primeiro e quando um veículo recebe uma mensagem duplicada ele cancela o seu reenvio. O DBRS tem problemas com o cancelamento desordenado nos reenvios, que pode ocasionar a falta de um encaminhador para mensagem.

Utilizando transmissão de vídeo para situações de emergência FELICE et al. (2015) propuseram um cenário, utilizando o simulador SUMO, aonde um veículo acidentado emite um alerta de segurança na rede. Esse alerta possui marcação geográfica e, utilizando dessa marcação, outros veículos que estão mais próximos do emissor do alerta, ligam as câmeras e começam a filmar. Os vídeos que estão sendo gravados serão enviados caso outros veículos da rede solicitem a transmissão. Dentro do cenário proposto pelos autores o vídeo também pode ser transmitido para um RSU para poder aumentar o área de transmissão. Nesse trabalho os autores utilizam a visão de apenas uma câmera que esteja mais próxima do acidente, o que diminui o campo de visão.

Outro trabalho de SMIDA; FANTAR; YOUSSEF (2018) propõe um aplicativo chamado *smart city's* (SMCVANET), esse aplicativo é responsável pelas interações V2V,V2I e V2X relacionados à transmissão de vídeos. Para melhorar a QoE na transmissão dos vídeos, foi implementado um protocolo *Delay Quality of Link and Link lifeTime aware routing protocol suitable*

for Video streaming (DQLTV). As transmissões de vídeos são classificadas de acordo com sua prioridade, situações de emergência tem a maior prioridade, seguidos por vídeos de assistência e informação ao motorista e por último entretenimento. Os vídeos podem ser armazenados em *buffer*, o veículo que solicitar o vídeo pode receber de um vizinho próximo e não necessariamente do veículo que está gerando o vídeo. Cada nó envia seus dados, como informação do GPS, velocidade e direção, para os nós vizinhos. Os dados dos nós vizinhos são armazenados em tabelas, como também informações do GPS, velocidade e direção que são enviadas aos seus vizinhos. A qualidade do link dos vizinhos é medida, com essas informações o protocolo DQLTV busca a melhor rota para a transmissão dos vídeos. Utilizando o simulador de rede NS3, o DQLTV foi comparado com outros protocolos, *ETX metric within the AODV* (AODV-ETX) proposto por JEVTIĆ; MALNAR (2018). O protocolo DQLTV obteve resultados superiores em relação aos outros dois que foi comparado nas métricas de QoE, como o *Peack Signal to Noise Ratio* (PSNR), *Mean Opinion Score* (MOS) e *Structural SIMilarity* (SSIM). O trabalho busca melhorar a QoE melhorando a transmissão do vídeo, mas não busca melhorar a geração do vídeo.

O trabalho de SANTOSH (2019), sugere um conceito para utilizar o compartilhamento de imagens em redes veiculares. As imagens são usadas para informação de acontecimentos de trânsito e com isso visa melhorar a experiência de condução dos motoristas. O trabalho também sugere o uso de imagens panorâmicas, geradas de dois ou mais veículos para melhorar a visão dos acontecimentos. No trabalho do autor não foi explicado formas de compartilhar as imagens, foi explicado apenas como realizar a junção das imagens e sugerido alguns tipos de aplicações para esse uso.

Atualmente não existe um consenso no setor automobilístico para medir a qualidade de imagens automotivas. O grupo IEEE P2020 está buscando corrigir essa deficiência, identificando as lacunas nos processos existentes, criando um padrão para análise de qualidade e conectando trabalhos que já estão em desenvolvimento para preencher essas lacunas (GROUP, 2018). O IEEE P2020 atualmente em desenvolvimento, visa padronizar: i) requisitos de qualidade de imagem e padrões de especificação; ii) padrões de oscilação de LED; iii) qualidade de imagem para visualização; iv) qualidade de imagem para visão computacional; v) interface do subsistema de câmera; vi) segurança de qualidade de imagem; vii) percepção do cliente sobre a qualidade da imagem.

4 REG3M (REENCAMINHADOR GEOGRÁFICO DE MENSAGENS MULTIMÍDIA)

Usar vídeos para informar situações de congestionamento se torna um problema para transmissão, pois é necessário enviar a mensagem para todos os envolvidos tentando evitar ou diminuir os problemas da *broadcast storm*. Alguns meios de diminuir os problemas da *broadcast storm* foram citados no capítulo 3, porém esses meios não foram pensados para mensagem maiores, como vídeo. Por isso, para a transmissão do vídeo foi criado o protocolo `ReG3M` (Reencaminhador Geográfico de Mensagens Multimídia), ele foi pensado para situação de mensagens de tamanho maiores que as mensagens comuns de alerta. O ReG3M busca encontrar o veículo mais distante para reencaminhar a mensagem, com isso ele diminui o número de reencaminhadores e mensagens transmitidas ao mesmo tempo.

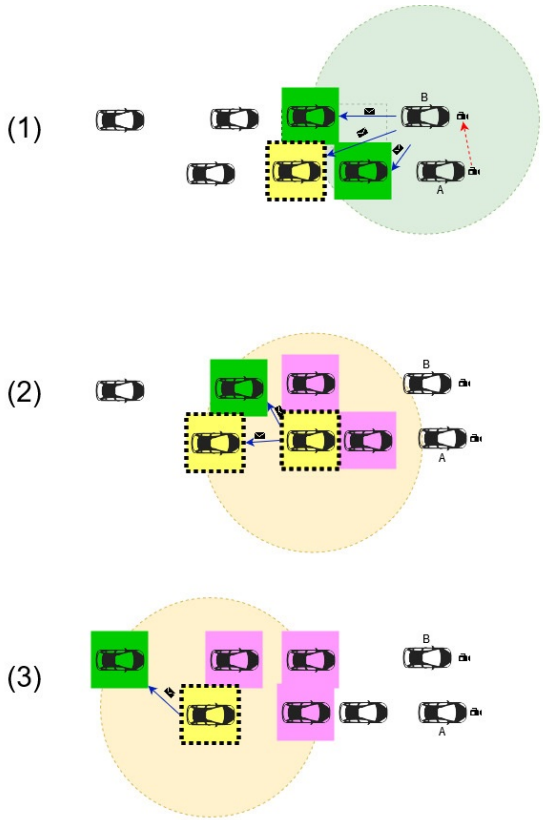
A Figura 4.1 ilustra o funcionamento do ReG3M, os veículos *A* e *B*, são os responsáveis por gravar os vídeos. No primeiro passo (1), o veículo *A* envia o vídeo para *B* que faz a união dos dois vídeos. Após isso o veículo *B* envia o vídeo gerado por homografia para todos que estiverem em seu alcance. Os veículos na cor verde, receberam a mensagem pela primeira vez. O veículo na cor amarela foi escolhido como o encaminhador da mensagem. No segundo passo (2), o veículo que foi escolhido como encaminhador transmite a mensagem com o vídeo para todos em seu alcance e escolhe o próximo encaminhador. Os veículos na cor rosa representam quem já recebeu a mensagem, ele descartam a mensagem por ser repetida. O terceiro passo (3), é uma repetição do passo anterior, isso ocorrerá até que todos os veículos do congestionamento tenham recebido a mensagem com vídeo.

4.1 Funcionamento do ReG3M

O protocolo ReG3M utiliza a distância entre o veículo de origem e destino da mensagem para escolher qual veículo é o melhor reencaminhador. O veículo de origem da mensagem faz uma comparação entre todas as distância dos veículos que receberam a mensagem, após isso elege o veículo mais distante como reencaminhador. Ao usar o veículo mais distante como reencaminhador o ReG3M consegue aumentar a distância percorrida pela mensagem. O ReG3M também controla as mensagens repetidas na rede, descartando mensagens que tenham a mesma identificação (*Message ID*) entre as mensagens já recebidas.

As regras do ReG3M buscam diminuir os problemas de *broadcast storm* reduzindo o número de mensagens na rede e o número de veículos retransmitindo, sem perder a eficiência em relação ao número de veículos que recebem a mensagem, distância percorrida pela mensagem

Figura 4.1 – Ilustração do ReG3M



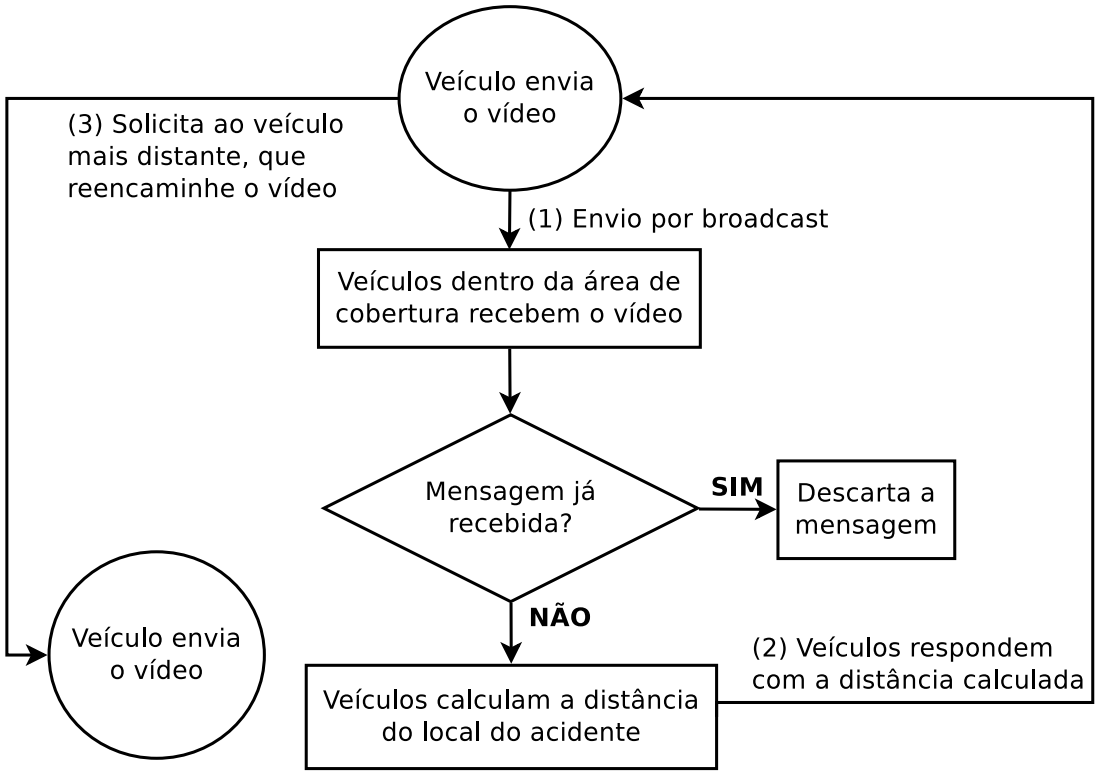
Fonte: do Autor

e velocidade de transmissão. A Figura 4.2 mostra o funcionamento do `ReG3M`, no qual o veículo que tem o vídeo completo faz a transmissão em *broadcast* (1) para os veículos que estão dentro do seu raio de alcance. Os veículos que recebem o vídeo completo respondem com a distância que estão do acidente (2). Então, o veículo que fez a transmissão irá eleger o mais distante para fazer o encaminhamento do vídeo completo (3). Após eleito, o veículo mais distante fica responsável por reencaminhar o vídeo completo. O processo se repete até que todos os veículos recebam a mensagem.

O protocolo ReG3M é implementado na camada de aplicação do *framework Veins*. A transmissão no ReG3M inicia assumindo que o veículo *B* tenha recebido o vídeo de *A* e realizado o processo de homografia. O veículo *B* cria uma mensagem com o vídeo e faz o envio em *broadcast*, após o envio a mensagem é inserida na sua lista de mensagens `MsgRec`. A mensagem enviada possui alguns campos importantes para as regras do protocolo ReG3M:

- `setSenderAddress(myId)`, possui o ID do veículo *B*;
- `setTipo(MSG_T_VIDEOCOMPLETO)`, tipo de mensagem que indica que o vídeo passou pelo processo de homografia;

Figura 4.2 – Funcionamento do ReG3M



- `setMsgId`, o ID da mensagem;
- `setPosicao`, recebe a posição geográfica do veículo.

Todos os veículos que recebem a mensagem de B verificam as mensagens recebidas. O Algoritmo 1 mostra as condições para o tipo de mensagem recebida. Quando a mensagem recebida é do tipo `MSG_VIDEOCOMPLETO` o veículo verifica se ela ainda não foi inserida em sua lista de mensagens, sendo uma mensagem nova o veículo irá enviar um *beacon* para o veículo de origem da mensagem contendo a distância entre eles.

Algorithm 1: Mensagem recebida pelos veículos

```
1 if (Mensagem->getTipo() == MSG_T_VIDEOCOMPLETO) then
2     if (msgRec.find(Mensagem->getMsgId()) == msgRec.end()) then
3         distancia = Mensagem->getPosicao() - mobility->
          getPositionAt(simTime()).length();
4         MaiorDistancia * Mensagem_Beacon = newMaiorDistancia;
5     end
6 end
```

A partir do momento em que o veículo de origem receber a primeira *Mensagem_Beacon*, com a distância do destino, ele irá criar um agendamento para aguardar 0,5 segundos. Esse

tempo é necessário para que todos os veículos que receberam o vídeo possam enviar as suas distâncias. Ao receber todas as distâncias o veículo origem calcula qual o veículo mais distante dele. Após o cálculo o veículo origem vai enviar um *beacon* ao destino, essa *beacon* recebe o tipo `BSM_T_FORWARD`. O veículo que recebe a `BSM_T_FORWARD` passa a ser o encaminhador do vídeo. O Algoritmo 2 mostra que ao receber o *beacon*, o veículo irá criar uma nova mensagem do tipo MSG_VIDEOCOMPLETO mantendo o ID da mensagem original.

Algorithm 2: BSM FORWARD

```
1 if (Mensagem_Beacon− > getTipo() == BSM_T_FORWARD) then
2 |   TraCIDemo11pMessage ∗ MensagemVideoCompleto =
  |     newTraCIDemo11pMessage;
3 end
```

Após o reenvio, os veículos que já tinham a mensagem, descartam o recebimento. O veículo que reencaminha o vídeo, recebe as distâncias de quem recebeu sua mensagem e o processo se repete, até que todos os veículos tenham recebido o vídeo.

5 METODOLOGIA

A metodologia para o desenvolvimento deste trabalho foi dividida em duas partes: Simulação de transmissão das mensagens utilizando o ReG3M, AID, DBRS e *Broadcast* e, processamento de imagens para a criação do vídeo através da homografia. Alguns parâmetros foram assumidos para a realização dos testes: o veículo que inicia a transmissão é o primeiro logo após o acidente e, os veículos iniciaram a gravação dos seus vídeos ao mesmo tempo para garantir a sincronia dos *frames*.

5.1 Simuladores

Para a análise das transmissões de mensagens nas redes veiculares foi utilizado o simulador OMNeT++ (*Objective Modular Network Testbed in C++*) (OMNET++, 2020). A simulação do tráfego veicular foi desenvolvida no SUMO (*Simulation of Urban Mobility*), capaz de identificar veículos, pedestres, transporte público e outros (FERRONATO; TRENTIN, 2015). Para a integração desses simuladores, foi utilizado o *framework* Veins (*Vehicular in Network Simulation*), que implementa os protocolos para redes veiculares estabelecidos pelo padrão IEEE 1609 (SOMMER; GERMAN; DRESSLER, 2011).

5.1.1 OMNeT++ (*Objective Modular Network Testbed in C++*)

O OMNeT ++ é um simulador $C++$ para redes de computadores, extensível, modular e baseada em componente. Com ele é possível simular redes de comunicação com e sem fio, redes de sensores, redes *ad hoc* sem fio, protocolos da internet, etc. O OMNeT++ oferece um IDE baseada em Eclipse. Muitos projetos independentes foram baseados no OMNeT++ o que aumenta a variedade de tipos de simulação que ele abrange. O OMNeT++ é distribuído sob licença pública para uso acadêmico (OMNET++, 2020).

5.1.2 SUMO (*Simulation of Urban Mobility*)

O SUMO é um simulador de trafego urbano, permite simular como uma determinada demanda de tráfego se comportaria. Foi desenvolvido para apoiar a comunidade de pesquisa de tráfego com uma ferramenta com a capacidade de implementar e avaliar algoritmos próprios. Cada veículo tem seu próprio trajeto e comportamento na rede. Permite a criação de mapas ou o uso de mapas reais em suas simulações (SUMO, 2020).

5.1.3 Veins

O Veins é um *framework* que faz a integração entre o SUMO e o OMNeT++ para realizar simulações de redes veiculares. O OMNeT++ faz as simulações de rede, a movimentação dos seus nós são reflexo dos movimentos gerados pelo SUMO. Veins tem origem de um projeto de pesquisa com o mesmo nome, visando uma avaliação de desempenho simulada aprimorada de redes veiculares. Enquanto esse projeto de pesquisa já está concluído, a estrutura de simulação Veins, parte dele desde o início de 2006, ainda está em desenvolvimento ativo como software de código aberto (SOMMER; GERMAN; DRESSLER, 2011).

5.2 Cenários

Para a simulação de transmissão do vídeo, foram criados três cenários para representar situações que ocorrem congestionamento no trânsito em uma cidade e uma rodovia.

- O primeiro com uma rodovia contendo duas faixas em um mesmo sentido. Dois veículos param, simulando um acidente que impede o trânsito nas duas faixas;
- O segundo cenário é um Grid 4x4, nele foi simulado a ocorrência de dois acidentes simultâneos, os dois acidentes acontecem em pistas separadas com uma distância aproximada de dois quilômetros entre eles. Cada um dos acidentes ocorrem em ruas contendo duas faixas em um mesmo sentido;
- Um terceiro cenário foi criado para avaliar o tempo que a mensagem demora para percorrer distância de 100, 200, 300, 400, 500, 600, 700, 800 metros. Esse cenário é semelhante ao primeiro, com duas faixas em um mesmo sentido.

Nos dois primeiros cenários foram executados os algoritmos ReG3M, AID, DBRS e envio por *broadcast*, para obter os resultados das seguintes métricas:

- **Distância percorrida -** Distância medida da origem da mensagem até o veículo mais distante que recebeu à recebeu;
- **Número de veículos alcançados -** Número total de veículos que receberam a mensagem;
- **Relação entre número de veículos e encaminhadores -** Número de encaminhadores que foi preciso para alcançar o número total de veículos, para os algoritmos que possuem controle no reencaminhador de mensagem (ReG3M, AID e DBRS);

- **Número de colisão na rede -** Número total de colisões nas transmissões da mensagem.

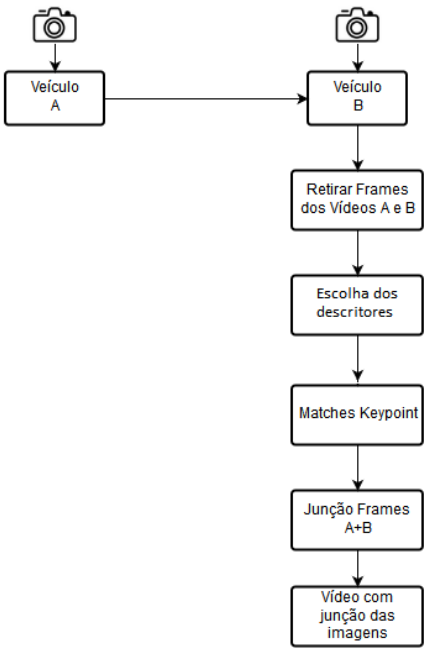
5.3 Criação do vídeo

Para a criação do vídeo foi utilizado a biblioteca OpenCV (*Open Source Computer Vision Library*). É uma biblioteca de código aberto, que foi criada para melhorar o uso da visão computacional em produtos e aplicativos (BRADSKI, 2000). A biblioteca possui mais de 2.500 algoritmos, incluindo um número abrangente de algoritmos de visão computacional e aprendizagem de máquina (BRADSKI, 2000).

Os algoritmos usados neste trabalho para detecção e descrição dos *keypoints* dos *frames* foram o SIFT, SURF, BRISK e ORB. Para a busca das correspondências foi utilizado o BFMatch.

A Figura 5.1 mostra os passos para a criação do vídeo. O veículo A e B são responsáveis, cada um, por gravar um vídeo. O veículo A envia o vídeo para o B, que faz o processo para criar o vídeo com homografia. O veículo B retira os *frames* dos dois vídeos, após isso utiliza um dos descritores para a retirada dos *keypoints* dos *frames*. Os *keypoints* passam pelo processo de *Match* para buscar as correspondência entre as imagens. Os *frames* são unidos e após isso é gerado um novo vídeo com os estes *frames*.

Figura 5.1 – Criação vídeo com homografia



O primeiro passo utilizado para a criação do vídeo usando homografia, foi a extração dos *frames* dos dois vídeos, conforme mostra o Algoritmo 3. A variável `vidObj` foi utilizada para receber o vídeo da função `cv2.VideoCapture` e faz a leitura do vídeo indicado. Após receber o valor, `vidObj` chama a função `read()` para extrair os *frames*.

Algorithm 3: Retirada dos *frames* dos vídeos

```
vidObj = cv2.VideoCapture(path)
while success do
    success, image = vidObj.read()
    cv2.imwrite(frame_path+name + "frame%d.jpg"% count, image)
end
```

O segundo passo é a junção dos *frames* dos dois vídeos, as imagens são lidas das duas pastas *A* e *B*. A cada iteração da estrutura de repetição, duas variáveis recebem as imagens da pasta *A* e *B*, respectivamente. O Algoritmo 4 mostra os procedimentos para encontrar a homografia dos *frames*. Após armazenar as imagens nas duas variáveis é necessário a criação do descritor. Neste trabalho são realizados testes com os descritores SIFT, SURF, ORB e BRISK.

Algorithm 4: Encontrar os *keypoints*/ Classificar e encontrar a homografia

```
descritor = cv2.xfeatures2d.SIFT_create() # ou SURF,ORB e BRISK
kp1, des1 = descritor.detectAndCompute(img1,None)
kp2, des2 = descritor.detectAndCompute(img2,None)
match = cv2.BFMatcher()
matches = match.knnMatch(des1,des2,k=2)
good = []
for m,n in matches do
    if m.distance < 0.5n.distance then
        good.append(m)
    end
end
MIN_MATCH_COUNT =10
if len(good) > MIN_MATCH_COUNT then
    src_pts = np.float32([ kp1[m.queryIdx].pt for m in good ]).reshape(-1,1,2)
    dst_pts = np.float32([ kp2[m.trainIdx].pt for m in good ]).reshape(-1,1,2)
    M, mask = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5)
end
```

Após a escolha do descritor, é feita a busca pelos *keypoints* das imagens. Os *keypoints* são encontrados pela função `descritor.detectAndCompute`, a busca é feita com o BFMatch nas duas imagens. A Figura 5.2 mostra os pontos com as correspondência das duas imagens. Com os *keypoints* encontrados é feito a busca por correspondências (*match*) nas duas imagens. Os resultado das buscas pelos *matches* devem ser classificados como "bons", quando possuem *matches* de vizinhos próximos.

Figura 5.2 – Pontos correspondentes nas 2 imagens

Com as "boas"correspondências das imagens encontradas, é realizada a busca pela homografia das imagens e junção das duas em uma imagem apenas. Para isso é usada a função `cv2.findHomography`, um dos parâmetros dessa função é o RANSAC, que encontra a matriz homográfica das imagens. Com essa matriz, é possível unir as imagens e corrigir a perspectiva caso estejam com ângulos diferentes conforme mostrado no Algoritmo 4. Todas as imagens geradas são salvas e usadas como *frames* para um novo vídeo. O resultado é uma imagem gerada da união de outras duas conforme mostra a Figura 5.3.

Figura 5.3 – Resultado junção das duas imagens

O terceiro passo é refazer o vídeo através das imagens que foram formadas no processo anterior. O vídeo é feito utilizando a função `cv2.VideoWriter` conforme mostrado do Algoritmo 5. A quantidade de *frames* por segundo do vídeo é um parâmetro da função e pode ser alterado de acordo com o que for necessário.

Algorithm 5: Criação do vídeo

```
1 pathInvideo = "Caminho dos frames"
2 out = cv2.VideoWriter(pathOutvideo,cv2.VideoWriter_fourcc(*'DIVX'), fps, size)
```

6 RESULTADOS E DISCUSSÕES

Este capítulo apresenta os resultados das simulações de transmissões do vídeo e a comparação do ReG3M com os algoritmos AID e DBRS e *broadcast*. São analisados cenários rodoviários e em grid quanto à variação do tamanho da mensagem, distância de transmissão e colisões. Em relação ao processamento de imagem, são analisados os descritores SIFT, SURF, BRISK e ORB, quanto à qualidade dos vídeos gerados para serem transmitidos na rede.

6.1 Criação do vídeo

Para a realização dos experimentos foram utilizados dois vídeos para cada teste. Para o primeiro teste dois vídeos de 5 segundos, com 30 *frames* por segundo, gravados em movimento e resolução de 1920x860. Para o segundo teste dois vídeos de 25 segundos, com 30 *frames* por segundo, gravados em movimento e resolução de 640x352. Para o terceiro teste dois vídeos de 60 segundos, com 30 *frames* por segundo, gravados em movimento e resolução de 640x352. Para melhorar a velocidade de processamento de junção dos *frames*, foram calculados os pontos característicos e os *match* apenas no primeiro *frame* de cada vídeo.

A homografia das imagens foi realizada utilizando os algoritmos SIFT, SURF, BRISK e ORB para a detecção de *keypoints*. Os *matches* foram feitos com o BFMatch. Os algoritmos utilizados para avaliar a qualidade de imagem foram o NIQE e o BRISQUE, eles são capazes de avaliar a qualidade da imagem utilizando apenas a própria imagem que está sendo avaliada (MITTAL; MOORTHY; BOVIK, 2012). O NIQE vária de 0 a 10, sendo 0 a melhor qualidade, enquanto que o BRISQUE, varia de 0 a 100, sendo 0 a melhor qualidade. Os valores da qualidade de imagem obtidos, representam o cálculo da média das qualidades de acordo com a quantidade de *frames* do vídeo.

Primeiro teste (vídeo de 5 segundos)

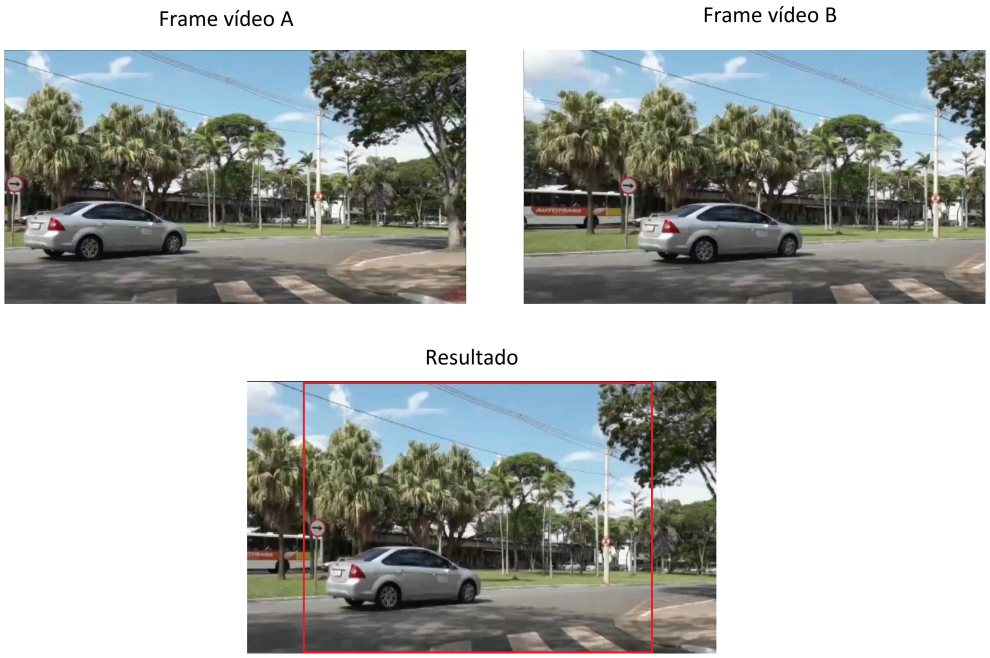
A Tabela 6.1 mostra os resultados obtidos utilizando cada um dos descritores. O descritor ORB foi o mais rápido em processamento na junção dos *frames*, sendo que a maior velocidade do ORB tem relação com a menor quantidade de *keypoints* que ele busca. Quanto aos índices de qualidade do NIQE e do BRISQUE, os algoritmos SIFT, SURF e BRISK obtiveram resultados muito próximos, porém tendo como destaque o BRISK que foi mais rápido em processamento e manteve a qualidade dos *frames* formados, mesmo tendo um aproveitamento de 34% na classificação dos *Good Matches*, valor menor que o SIFT e SURF.

Tabela 6.1 – Comparação dos descritores (vídeo 5s)

Descritor	**Tempo de processamento** (em segundos)	**Matches**	**Good Matches**	**NIQE** Índice de Qualidade	**BRISQUE** Índice de Qualidade
BRISK	12,782	4943	1729	3,010	27,628
ORB	12,042	500	149	3,115	28,361
SIFT	13,527	4916	2846	3,010	27,662
SURF	14,614	10234	5607	3,010	27,689

O resultado da junção dos *frames* pode ser visto na Figura 6.1, a marcação mostra as partes correspondentes entre as imagens:

Figura 6.1 – Junção *frames* (vídeo 5s)



Segundo teste (vídeo 25 segundos)

Em relação aos resultados obtidos no segundo teste a Tabela 6.2 mostra que mesmo o vídeo tendo uma duração maior que o primeiro teste, todos os descritores conseguiram um tempo reduzido de processamento, isso se deve ao fato que o vídeo com resolução menor tem menos *pixels* para ser analisados na busca por *keypoints*. A qualidade dos *frames* resultantes e a velocidade de processamento foi muito próxima em todos os descritores. Novamente o ORB foi o mais rápido e o que busca por menos *keypoint*, mas sua qualidade foi pior que os outros três algoritmos. O algoritmo SIFT mesmo tendo um aproveitamento de 51,22% dos seus *Macthes*

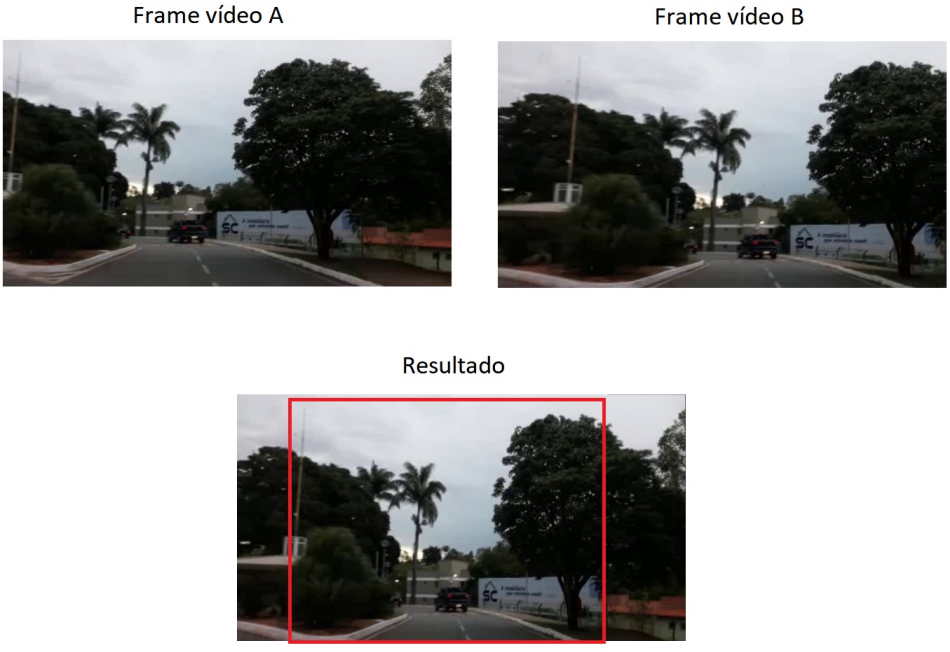
não conseguiu superar a qualidade do BRISK, que teve 36,9% de aproveitamento dos *Matches* em relação os *Good Matches*.

Tabela 6.2 – Comparação dos descritores (vídeo 25 segundos)

Descritor	**Tempo de processamento** (em segundos)	**Matches**	**Good Matches**	**NIQE** Índice de Qualidade	**BRISQUE** Índice de Qualidade
BRISK	13,736	1027	379	2,9158	28,459
ORB	12,296	500	114	2,953	29,002
SIFT	12,25	939	481	2,935	28,688
SURF	12,68	1074	531	2,962	28,487

O resultado da junção dos *frames* pode ser visto na Figura 6.2, a marcação mostra as partes correspondentes entre as imagens:

Figura 6.2 – Junção *frames* (vídeo 25s)



Terceiro teste (vídeo 60 segundos)

A Tabela 6.3 mostra os resultados com o vídeo de 60 segundos. Como a busca por pontos característicos e *match* foi realizada apenas no primeiro *frame*, o processo se tornou um pouco mais lento se for levar em conta a relação duração dos vídeos x tempo de processamento por

causa da diminuição dos pontos encontrados nos *frames* ao longo do vídeo. O ORB foi o mais rápido em processamento e a qualidade dos vídeos foi muita próxima entre todos os descritores. Em relação a qualidade o SURF obteve resultados melhores no NIQE e no BRIQUE, tendo $52,4\%$ de aproveitamento nos *Matches*. Entretanto o tempo de processamento do SURF foi $6,2$ segundos mais lento que o BRISK que teve qualidade muito próxima a ele. Levando em consideração Tempo *x* Qualidade o BRISK teve resultado melhor que o SURF.

Tabela 6.3 – Comparação dos descritores (vídeo 60 segundos)

Descritor	**Tempo de processamento** (em segundos)	**Matches**	**Good Matches**	**NIQE** Índice de Qualidade	**BRISQUE** Índice de Qualidade
BRISK	29,288	697	238	3,404	35,491
ORB	28,552	500	118	3,404	35,549
SIFT	37,409	605	302	3,405	35,594
SURF	35,49	771	404	3,395	35,215

O resultado da junção dos *frames* pode ser visto na Figura 6.3, a marcação mostra as partes correspondentes entre as imagens:

Figura 6.3 – Junção *frames* (vídeo 60s)

6.1.1 Análise dos experimentos

Analisando os resultados entre os três testes, pode-se observar uma diferença sútil na qualidade dos vídeos gerados por todos os descritores. Isso ocorre pois a partir do momentos

que junção dos vídeos é realizada com sucesso a qualidade dos vídeos originais se mantém. Caso a junção não ocorra da maneira correta, o *frame* que teve problemas ficará desfigurado e diminuirá a qualidade do vídeo. Nos três testes não ocorreu em nenhum momento este problema. Os algoritmos SIFT e SURF conseguem ter um aproveitamento, nos três testes, próximo à 50% em seus *Matches* sendo estes considerados *Good Matches*. Os *Good Matches* garantem o sucesso nas junção dos *frames*. Porém nos testes realizados neste trabalho, os algoritmos ORB e BRISK conseguem ter o sucesso mesmo tendo um menor aproveitamento dos *Matches*. Contudo levando em consideração os três testes realizados, o algoritmo BRISK teve a melhor relação entre Tempo *x* Qualidade.

6.2 Transmissão do vídeo

A transmissão do vídeo foi simulada no OMNeT++, considerando os 3 cenários citados no capitulo 5, onde os veículos estão congestionados em fila dupla. Os cenários utilizados foram considerados para situações, nas quais somente a fila formada atrás dos veículos que transmitem o vídeo sejam os interessados. Outros veículos que não estão neste caminho não teriam interesse.

Os experimentos de transmissão foram realizados comparando o ReG3M com os protocolos AID, DBRS, *Broadcast*, apesar de serem algoritmos que não utilizam a escolha de um *Cluster* para reenvio, devido ao fato de na literatura não ter sido encontrado algoritmos que utilizem *Cluster* para envios em *broadcast* em VANET.

Os resultados comparados nos 4 algoritmos foram, número de veículos que receberam a mensagem, distância percorrida pela mensagem, relação entre número de veículos e quantidade de encaminhadores (nos algoritmos ReG3M, AID, DBRS), número de colisões na rede e velocidade de transmissão da mensagem. Para obter os resultados foram realizadas 33 simulações com cada algoritmo de encaminhamento e com mensagens de 10 Mb, 5 Mb, 100 Kb e 10 Kb.

Os parâmetros utilizados nas simulações estão na Tabela 6.4. Todos os parâmetros desta tabela são fixos para todas as simulações, com exceção do `piso de ruído`, gerado aleatoriamente durante os testes.

Tabela 6.4 – Parâmetros da simulação

Parâmetro	**Valor**
Versão do OMNeT++	5.5.1
Versão do SUMO	1.2.0
Versão do Veins	5.0
Sensibilidade das antenas	-110dbm
Piso de ruído	aleatório: [-85dbm, -75dbm]
Potência do sinal	5mW
Taxa de transferência	6Mbps
Tempo de duração para envio da mensagem	120 segundos

Os algoritmos AID e DBRS fazem o controle de reenvio de mensagens baseados no em tempo de espera. Para nossa simulação foi necessário considerar o tamanho da mensagem nos tempos de espera dos dois algoritmos, para que fosse possível o recebimento da mensagem completa pelos veículos. Sem considerar o tamanho da mensagem, os dois algoritmos apresentaram comportamento semelhante ao *flooding*, com todos os veículos realizando os reenvios.

O tempo de espera do AID passou a ser a soma de um tempo aleatório T (vária entre 0,05 e 0,01) mais o resultado do tamanho da mensagem $TamanhoMsg$ dividido pela taxa de transferência $TaxaT$ da simulação. A equação 6.1 mostra o cálculo:

$$T + \frac{TamanhoMsg}{TaxaT} \tag{6.1}$$

O Algoritmo 6 mostra as regras do algoritmo AID. As linhas 1 e 2 mostram a criação de um agendamento, que representa o tempo de espera que cada veículo tem para processar se as mensagens recebidas são repetidas ou não. O tempo de espera é o tempo de simulação mais T. As linhas 3 até 5 mostram como é realizada a contagem e a frequência de mensagens repetidas. Enquanto o tempo de espera ainda não se encerrou, o algoritmo incrementa o contador c sempre que receber um mensagem repetida. Para cada mensagem repetida é calculado o tempo que a mensagem demorou da origem até o destino, esse tempo é inserido em um vetor ΔT. Após encerrar o tempo de espera será analisado se o veículo deve ou não ser um retransmissor. O retransmissor é mostrado nas linhas 6 até 14, para definir o retransmissor será realizado uma verificação em todos os valores de ΔT. Se os valores de ΔT forem maiores que $\frac{t}{c}$ uma variável

s será acrecida, se o valor for menor s será decrementada. Ao final das comparações se o valor de s for maior ou igual a 0, o veículo irá realizar a retransmissão.

Algorithm 6: Cálculo para mensagens repetidas

```
   /* Agendamento de mensagem "SelfMsg"                                  */
 1 SelfMsg * selfMsg, myId
 2 scheuduleAt(simTime() + T, selfMsg)
   /* Contagem de mensagens repetidas                                    */
 3 c++
 4 tempo = simTime() - Mensagem-> getTempo()
 5 delta_t.push_back(tempo)
   /* Retransmissor AID                                                  */
 6 s = 0
 7 for (i = 0; i < c; i++) do
 8     if (delta_t [i] > (t/c)) then
 9         s++
10     end
11     else
12         s--
13     end
14 end
```

O tempo de espera do DBRS passou a ser 1 dividido pela distância d entre o veículo que recebe o vídeo e o veículo de origem, mais o resultado do tamanho da mensagem $TamanhoMsg$ dividido pela taxa de transferência $TaxaT$ da simulação. A equação 6.2 mostra o cálculo:

$$\frac{1}{d} + \frac{TamanhoMsg}{TaxaT} \tag{6.2}$$

As regras para retransmissão do DBRS são baseadas em um tempo inversamente proporcional à distância entre origem e destino da mensagem. Para execução da regra é necessário calcular a distância que o veículo que recebeu a mensagem está da origem da mensagem. Após isso, é feito o cálculo da equação 6.2, o resultado desse equação é o tempo que o veículo irá esperar para retransmitir a mensagem. Caso o veículo receba uma mensagem repetida, ele irá cancelar o reenvio que está esperando para realizar. O Algoritmo 7 mostra como é a execução

do DBRS:

Algorithm 7: Retransmissor DBRS

1 $distancia = (Mensagem- > getPosition() - mobillity- : getPosition(simTIme())$

2 $tempo = (1/distancia) + TamanhoMsg/TaxaT$

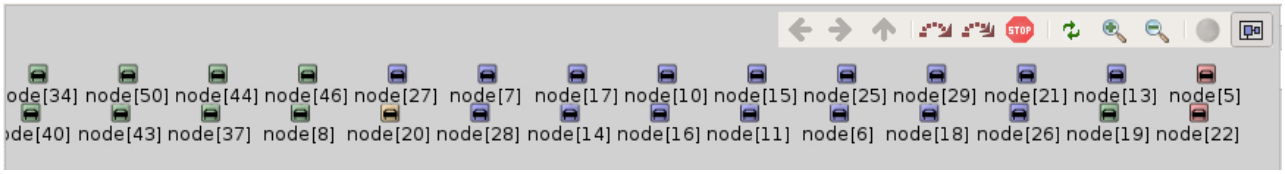
3 $SelfMsg * selfMsg = newSelfMsg$

4 $scheduleAt(simTime() + tempo)$

6.2.1 Cenário I - Rodovia

As simulações do primeiro cenário foram feitas para representar uma rodovia de faixa dupla. A Figura 6.4 mostra o trecho inicial da simulação, no qual os veículos em vermelho sofrem acidente. O `node[13]` envia a primeira parte do vídeo para o `node[19]`, que faz a homografia e envia o mesmo por *broadcast*. O veículo na cor amarela foi eleito para reencaminhar o vídeo para os demais. Os veículos verdes receberam o vídeo pela primeira vez, enquanto os azuis receberam o vídeo repetido e descartaram a mensagem.

Figura 6.4 – Trecho da simulação no OMNeT++



Os resultados do ReG3M, no cenário *I*, foram superiores em mensagens maiores devido ao fato que, os veículos ficam muito tempo ocupados até concluir o recebimento, como no ReG3M existe apenas um reencaminhador por vez, ele evita colisões na rede. O controle do *broadcast* no ReG3M foi superior que os outros algoritmos, com isso cada reencaminhador conseguiu atingir um número maior de veículos o que fez ser necessário um número menor de reencaminhadores para atingir mais veículos. O uso de apenas um reencaminhador também fez com que o número de colisões na rede do ReG3M fosse inferior aos outros algoritmos. Para mensagens de $5Mb$ o tempo gasto para o recebimento completo da mensagem é menor do que nas mensagens de $10Mb$, porém mesmo com o tempo menor, o ReG3M ainda segue superior aos outros algoritmos em todas as métricas analisadas. Mesmo em mensagens menores de $100Kb$ o ReG3M segue superior aos outros algoritmos em todas as métricas, porém em mensagens de $10Kb$ o ReG3M foi ligeiramente inferior aos algoritmo AID em número de veículos e distância

alcançada. O ReG3M se manteve melhor que todos os algoritmos testados em relação a número de colisões e e relação entre veículos e número de encaminhadores com todos os tamanhos de mensagens testados. Os gráficos a seguir mostram os resultados obtidos em testes com mensagens de 10Mb, 5Mb, 100Kb e 10Kb.

Teste com mensagens de 10 Mb

As Figuras 6.5 e 6.6 mostram, respectivamente, o número de veículos e a distância que a mensagem de 10Mb alcançou nos teste. Nelas podemos observar que o ReG3M obteve um resultado superior aos outros algoritmos, conseguindo alcançar aproximadamente 340% mais veículos e $1.055,10$ metros a mais que o algoritmo AID, segundo melhor colocado.

Figura 6.5 – Veículos alcançados em 120 segundos (10 Mb)

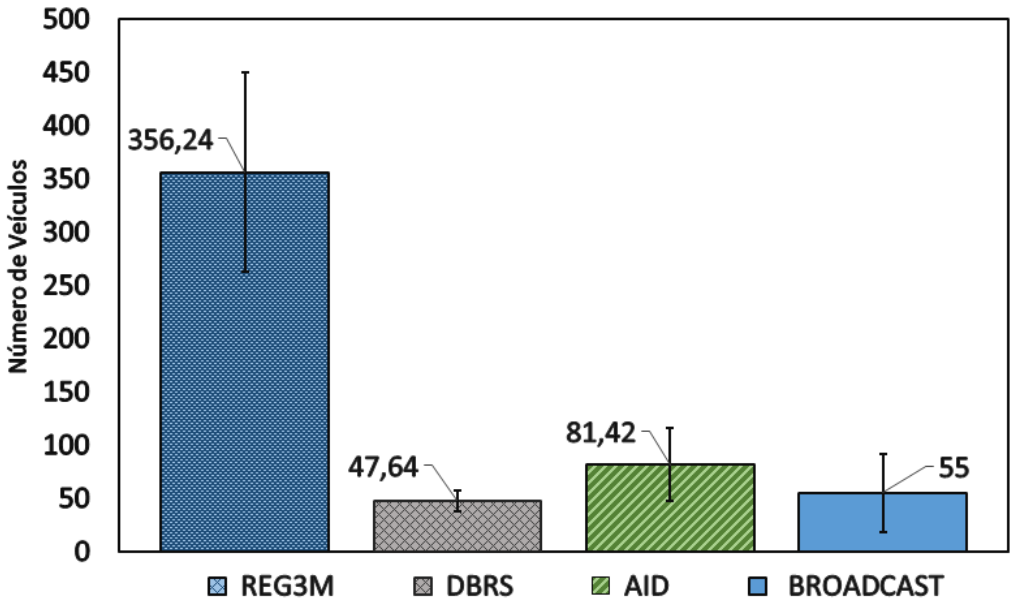
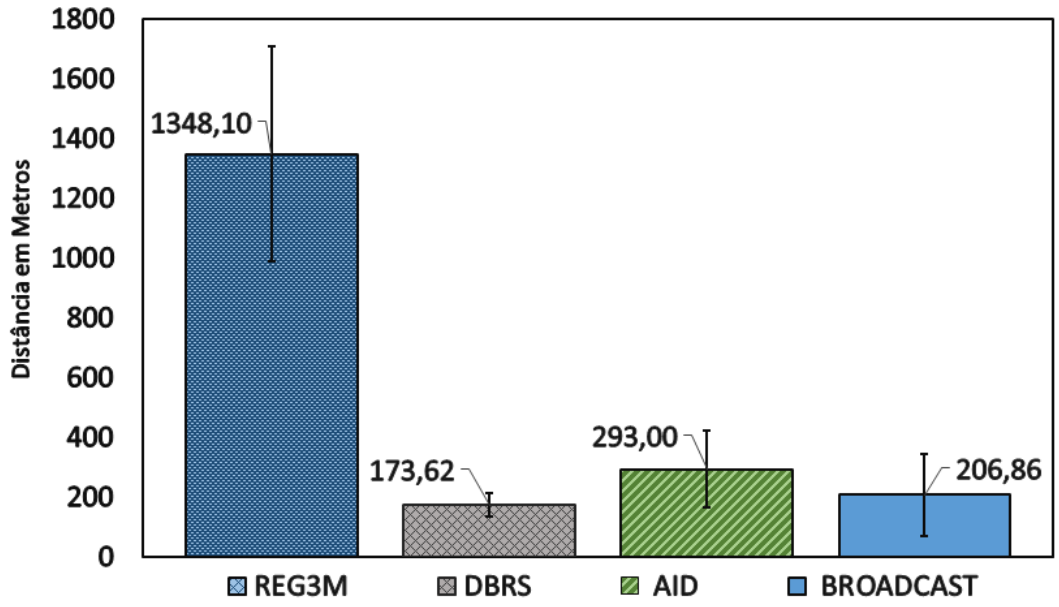
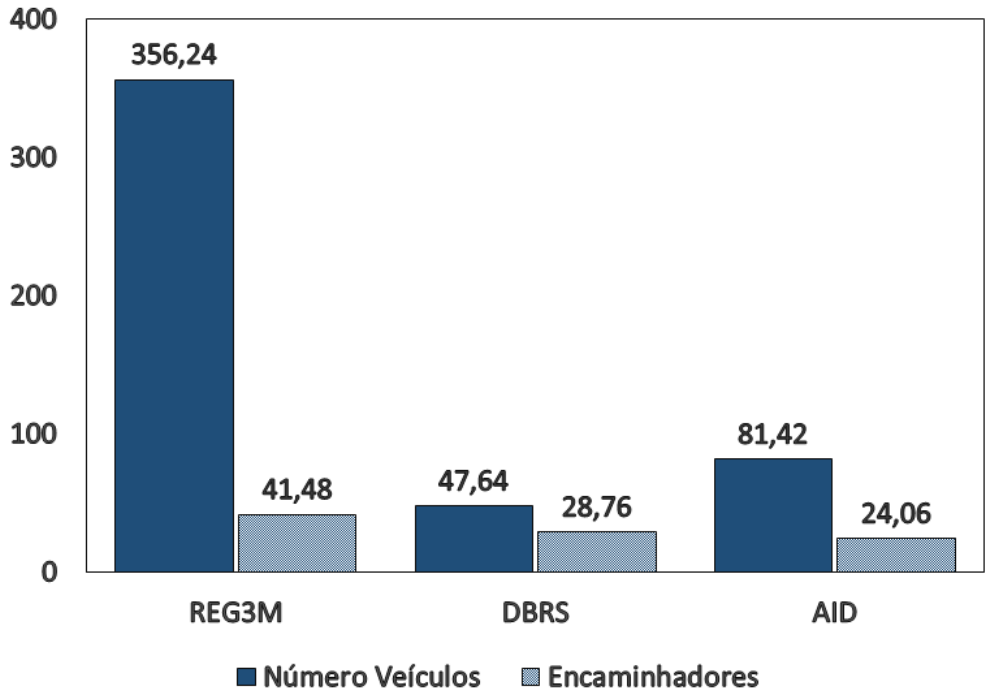


Figura 6.6 – Distância alcançada em 120 segundos (10 Mb)



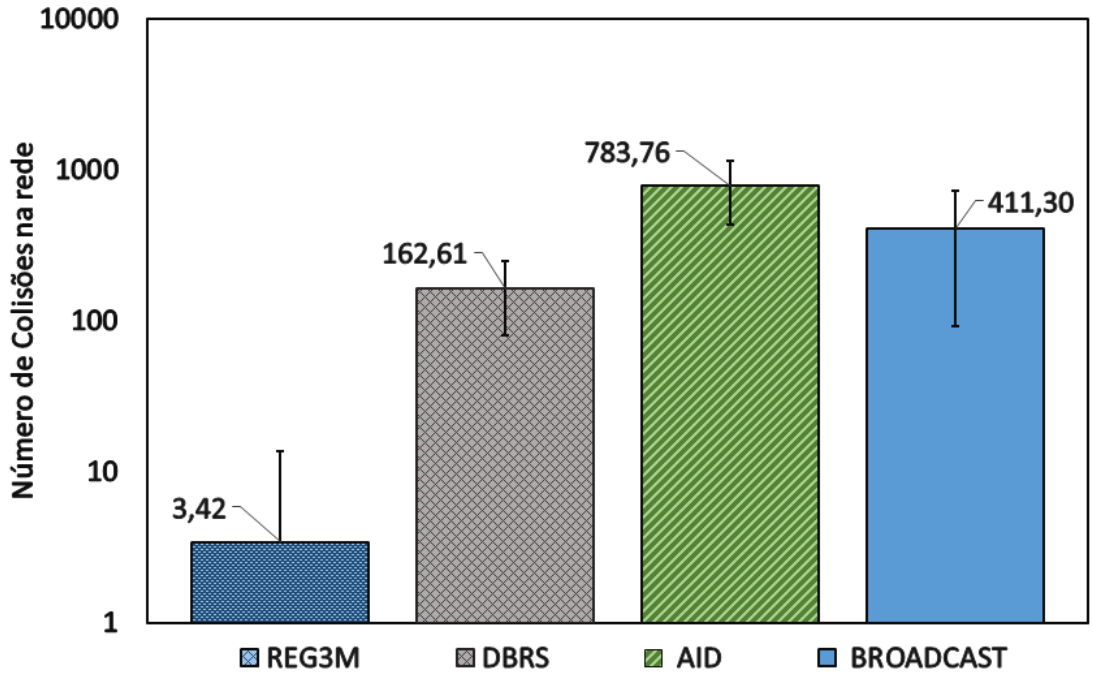
A Figura 6.7 mostra a relação dos veículos alcançados e o número de encaminhadores necessários para conseguir esse alcance. Os resultados são referentes aos três algoritmos que controlam o reenvio, AID, DBRS e ReG3M. O ReG3M conseguiu alcançar um número superior de veículos com um número pequeno de encaminhadores.

Figura 6.7 – Relação entre veículos alcançados e número de encaminhadores (10 Mb)



A Figura 6.8 mostra o número de colisões na rede. Como o controle de reenvio do ReG3M foi melhor e teve menos encaminhadores, os testes de colisões foi consideravelmente menor do que o dos outros algoritmos, mesmo com ele alcançando um número maior de veículos.

Figura 6.8 – Número de colisões na rede (10 Mb)



Testes com mensagens de 5 Mb

Nos testes de $5Mb$ podemos ver uma melhora no alcance da mensagem em número de veículos e distância em todos os algoritmos como mostrados nas Figuras 6.9 e 6.10. O ReG3m ainda é superior aos outros algoritmos, seu percentual de veículos alcançados é aproximadamente 353% maior que o AID. O ReG3M também teve $1.570,21$ metros a mais de distância percorrida pela mensagem que o segundo colocado.

Figura 6.9 – Veículos alcançados em 120 segundos (5 Mb)

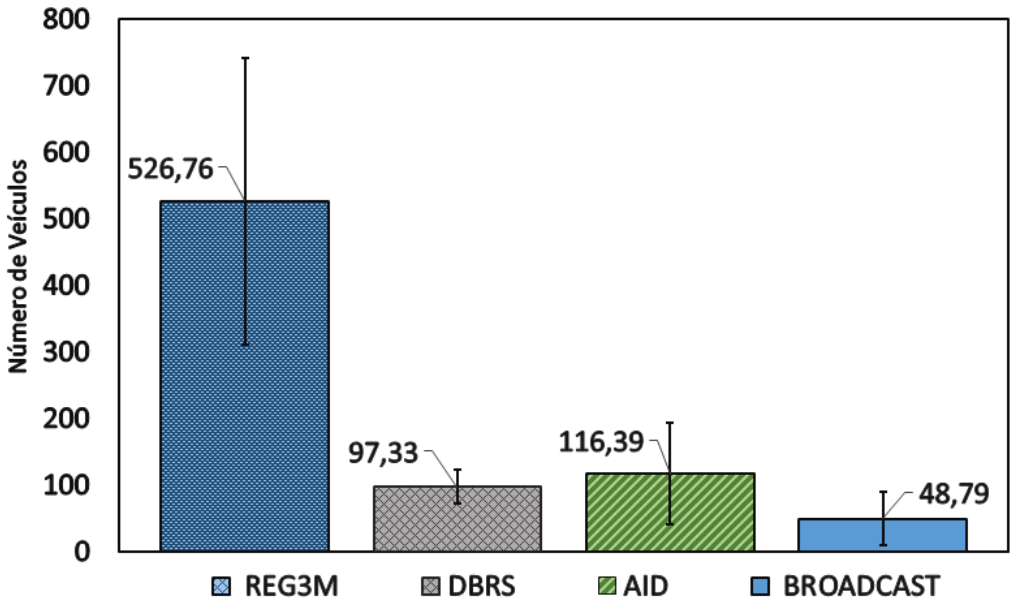
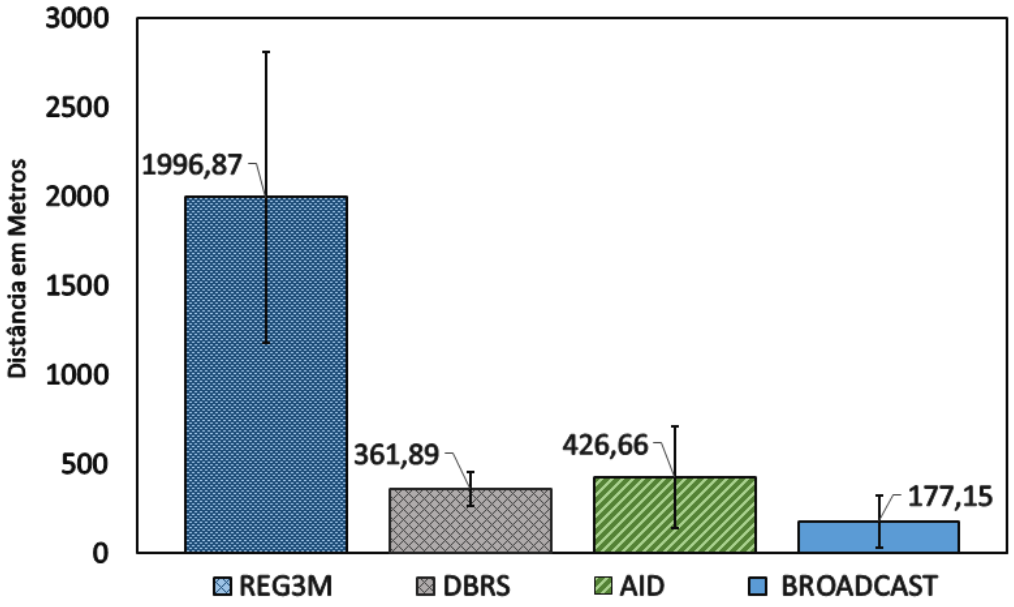
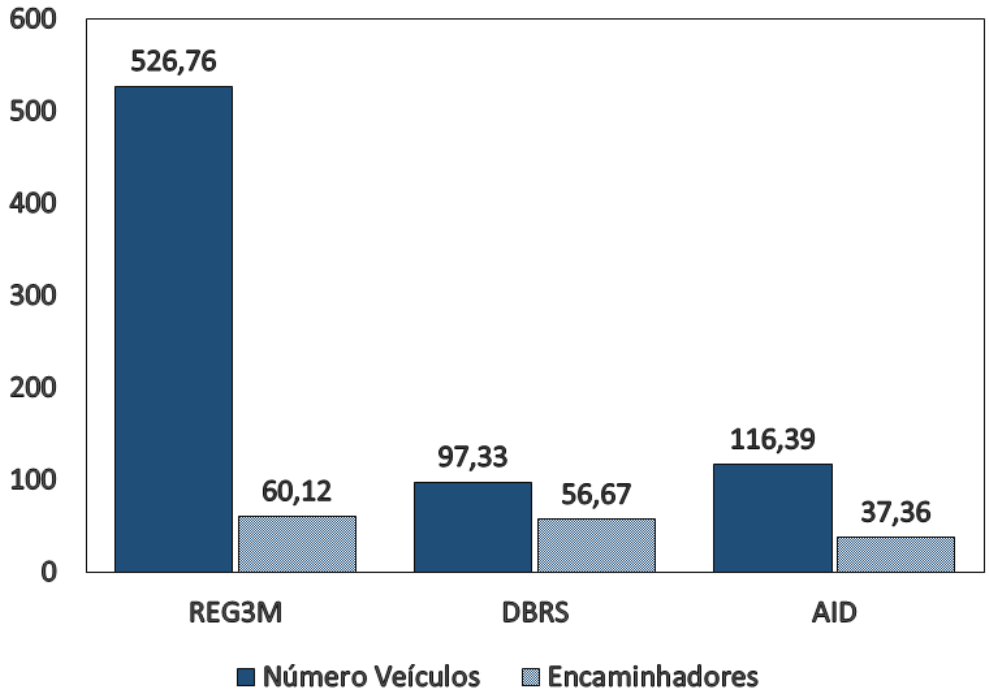


Figura 6.10 – Distância alcançada em 120 segundos (5 Mb)



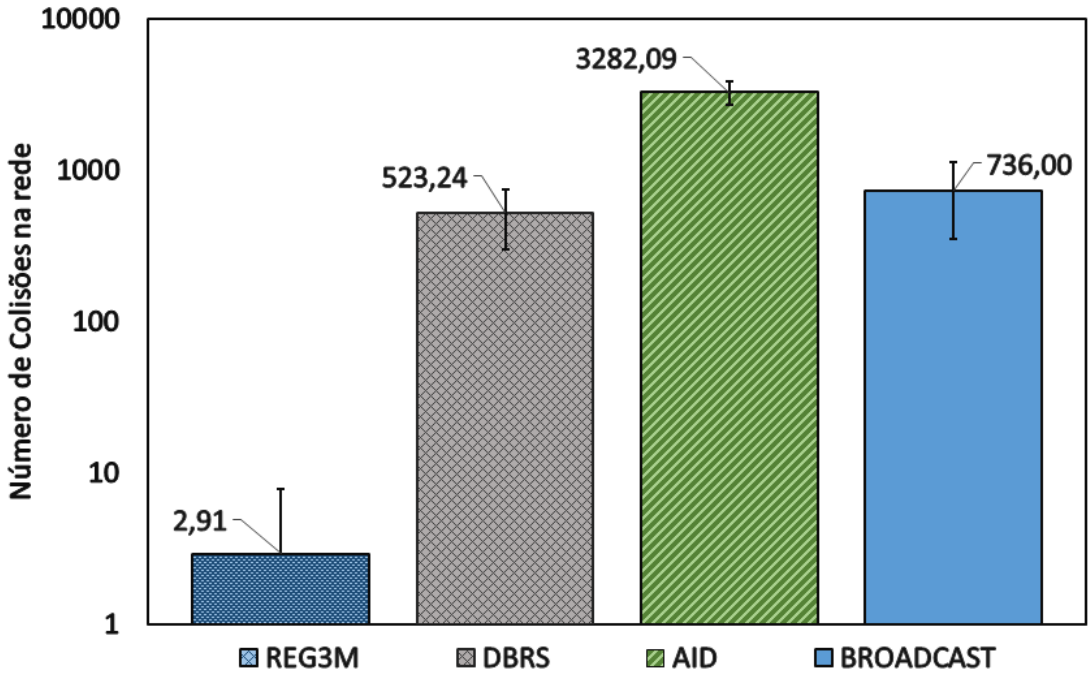
Em relação ao número de encaminhadores para os teste com mensagens de 5*Mb* o ReG3M também conseguiu resultados melhores que os outros dois algoritmos, obtendo um número mais alto de veículos com um número proporcionalmente menor de encaminhadores. Conforme mostra os resultados na Figura 6.11.

Figura 6.11 – Relação entre veículos alcançados e número de encaminhadores (5 Mb)



Em relação ao número de colisões na rede, conforme mostra a Figura 6.11, o ReG3M continua com o número muito menor do que os outros algoritmos. O número de colisões em todos os algoritmos aumentou devido ao fato do número de veículo alcançado ter sido maior em todos os testes.

Figura 6.12 – Número de colisões na rede (5 Mb)



Teste com mensagens de 100 Kb

Nos testes com $100Kb$, conforme mostra as Figuras 6.13 e 6.14, o ReG3M continua em média superior aos outros algoritmos, tendo alcançado aproximadamente 93% veículos a mais que o AID. A mensagem percorreu aproximadamente $1.771,82$ metros a mais que o segundo colocado.

Figura 6.13 – Veículos alcançados em 120 segundos (100 Kb)

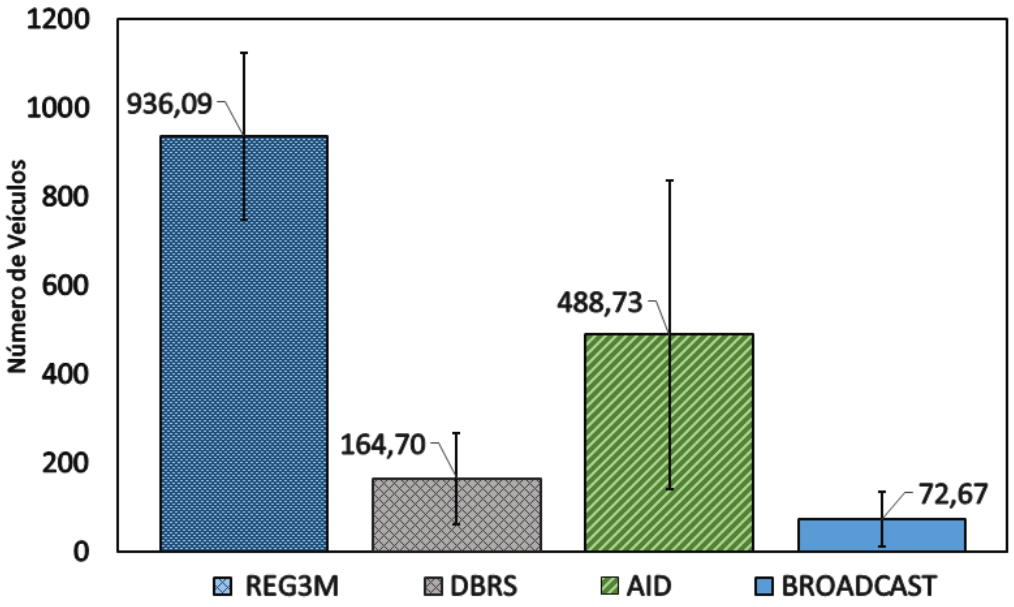
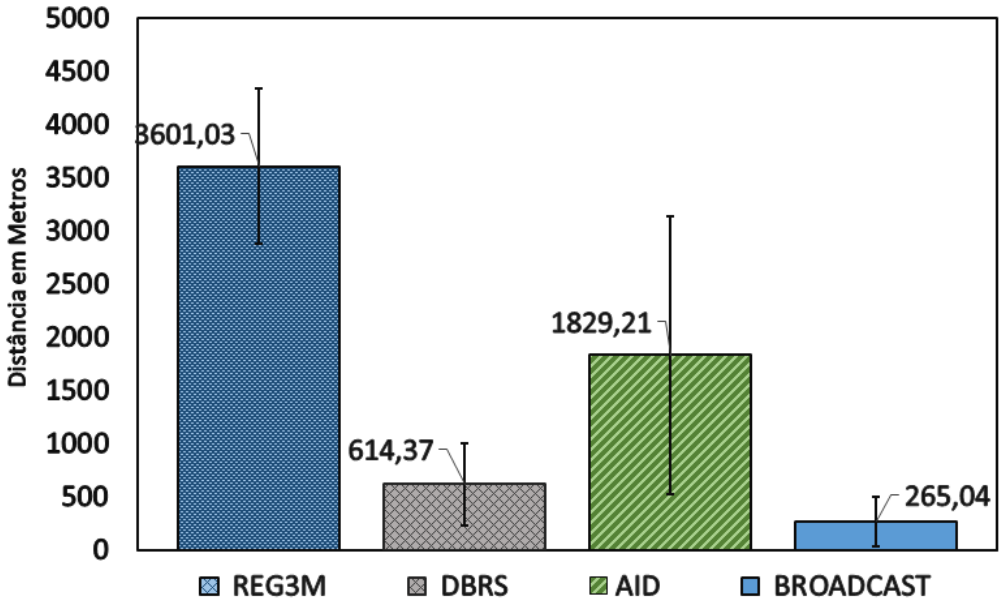


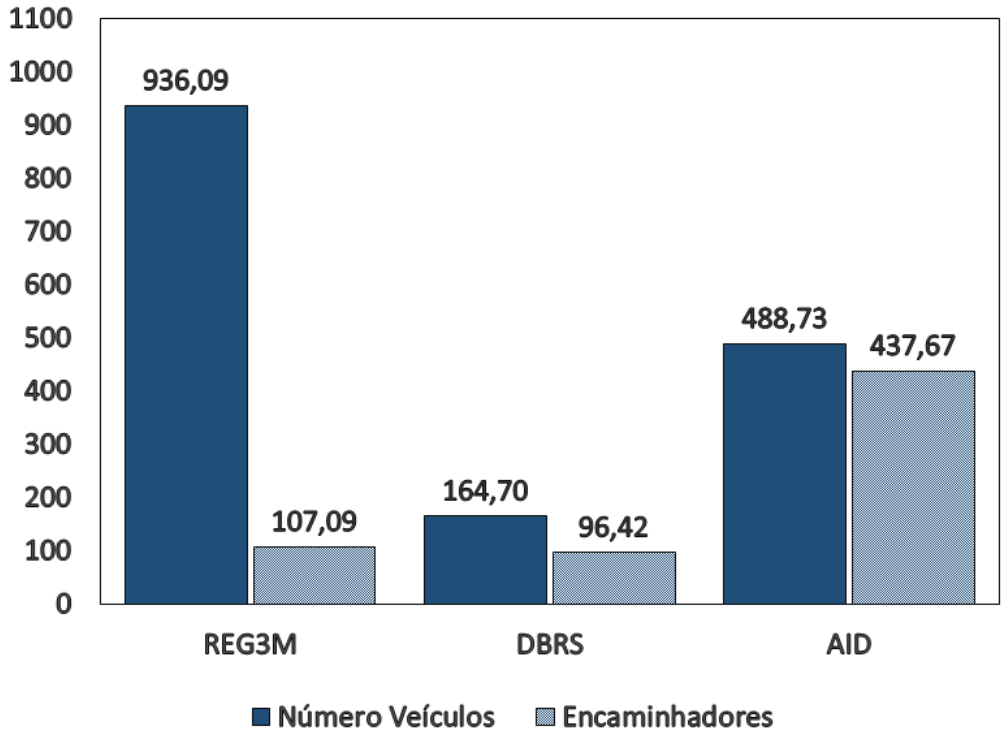
Figura 6.14 – Distância alcançada em 120 segundos (100 Kb)



O número de encaminhadores do ReG3M continua baixo proporcionalmente ao seu alcance. O AID que conseguiu resultados de distância e número de veículos melhores do que o

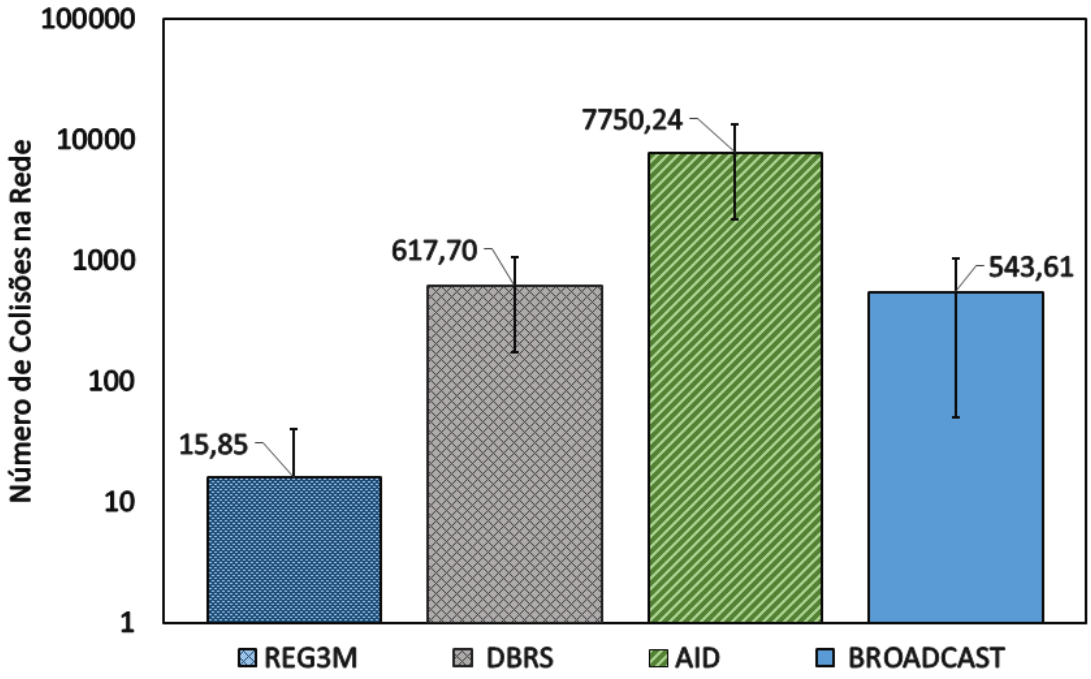
ReG3M, em alguns testes, teve um aumento significativa na proporção dos seus encaminhadores, conforme mostrado na Figura 6.15

Figura 6.15 – Relação entre veículos alcançados e número de encaminhadores (100 Kb)



O ReG3M continua sendo o algoritmo com menor número de colisões na rede e seus resultados mostram uma diferença muito grande para os outros algoritmos. O AID foi o algoritmo com mais colisões, porém conseguiu uma alcance melhor que os outros com exeção do ReG3M. A Figura 6.16 mostra os resultados.

Figura 6.16 – Número de colisões na rede (100 Kb)



Teste com mensagem de 10 Kb

Nos testes de $10Kb$ o ReG3M foi superado pelo algoritmo AID e ficou a frente do DBRS e *broadcast* não controlado, em relação ao alcance de número de veículos e distância. Conforme mostra as Figuras 6.17 e 6.18. O AID foi ligeiramente melhor e teve 12% de veículos e aproximadamente 404 metros a mais que o ReG3M. Isso se deve ao tempo $0,5s$ de espera do ReG3M para recebimento dos *beacons* com as distâncias dos veículos. Para mensagens menores é mais rápido reenviar a mensagem do que esperar para controlar o número de reenvio.

Figura 6.17 – Veículos alcançados em 120 segundos (10 Kb)

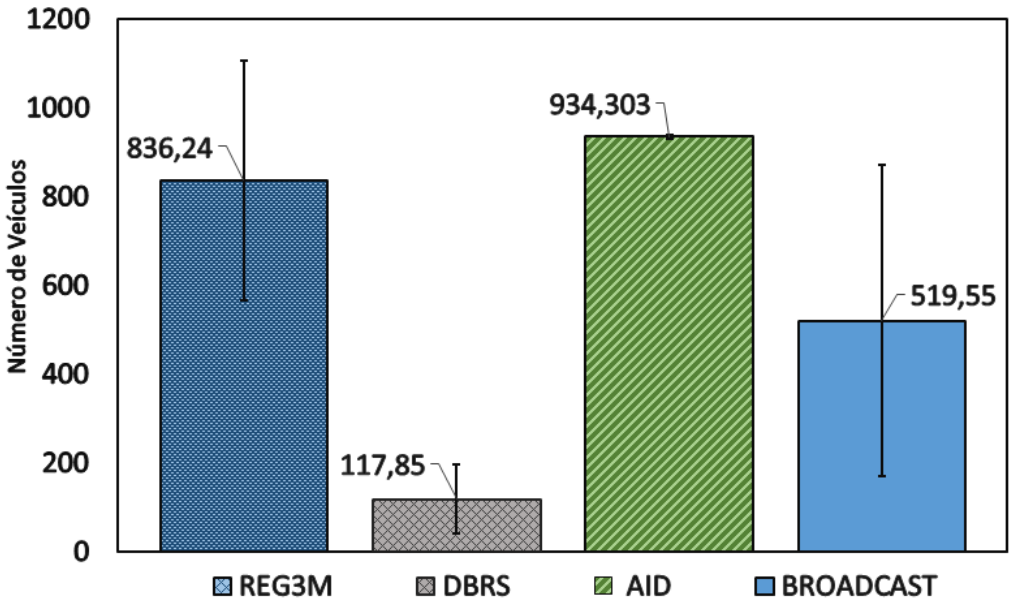
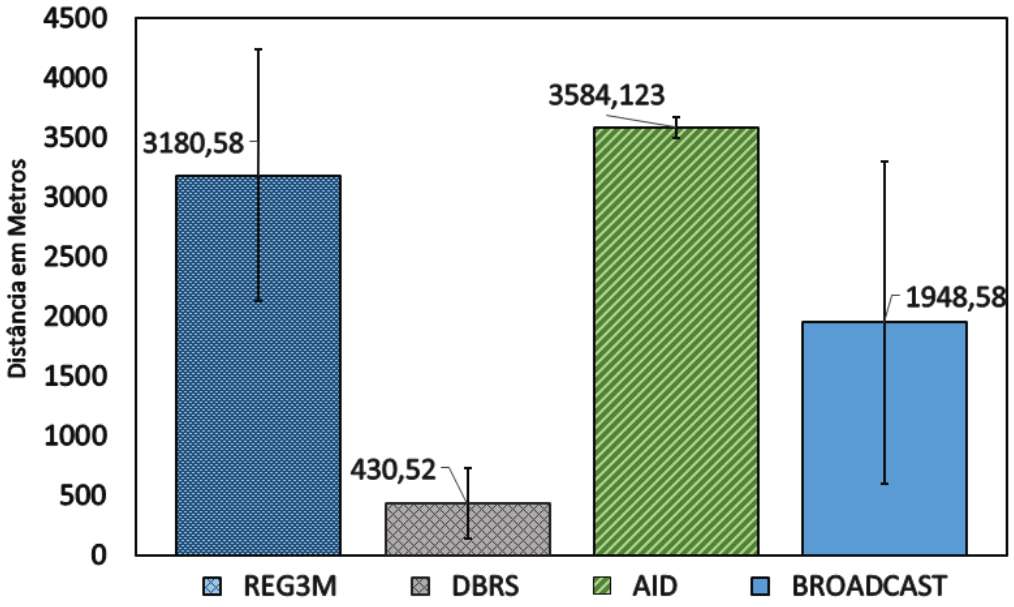
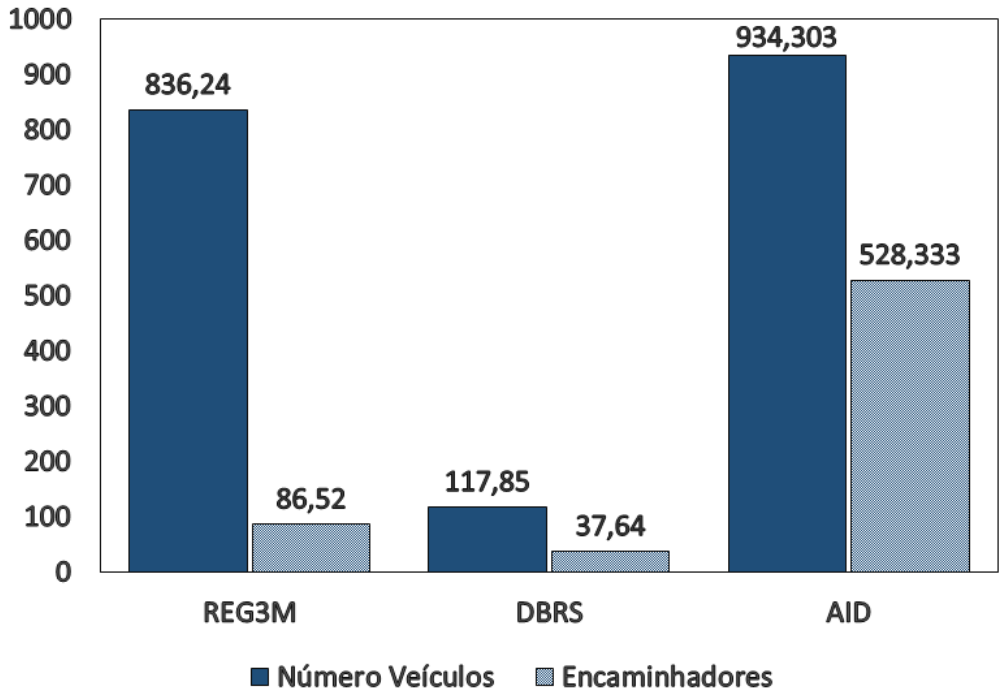


Figura 6.18 – Distância alcançada em 120 segundos (10 Kb)



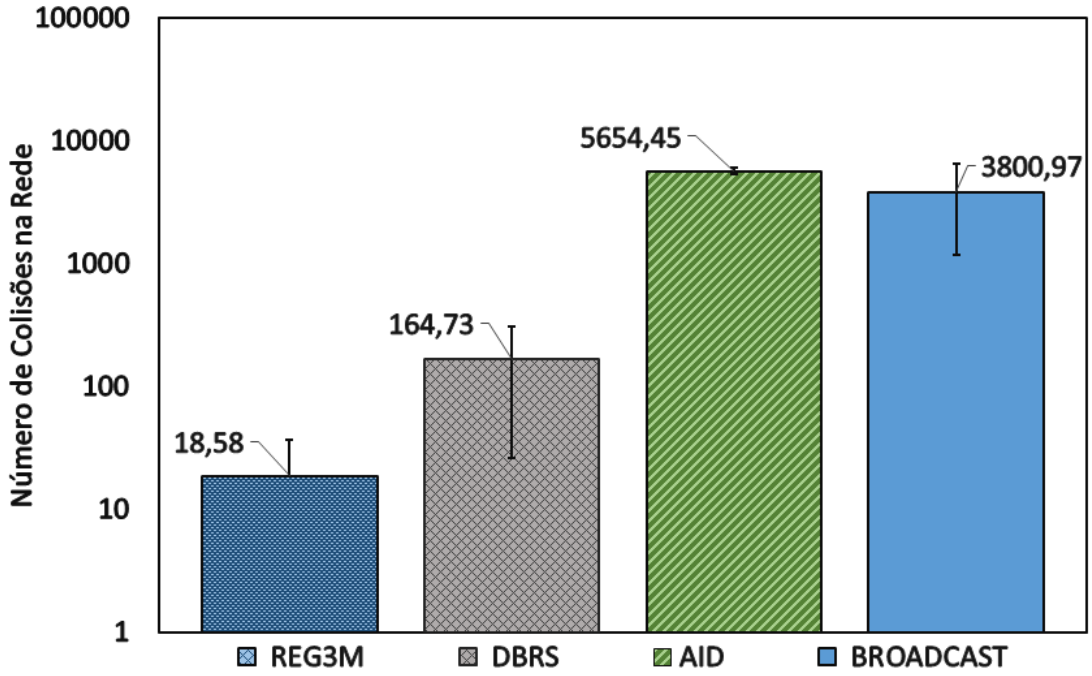
A Figura 6.19 mostra que o número de encaminhadores do ReG3M ainda é menor que dos outros algoritmos, mostrando que ele consegue bons resultados mesmo com mensagens menores.

Figura 6.19 – Relação entre veículos alcançados e número de encaminhadores (10 Kb)



O número de colisões do ReG3M é consideravelmente menor do que os outros algoritmos mesmo tendo resultados proximos ao AID em relação a distância e número de veículos. O AID obteve elevado número de colisões devido ao alcance dele ter sido maior do que a dos outros algoritmos. A Figura 6.20 mostra os resultados.

Figura 6.20 – Número de colisões na rede (10 Kb)



6.2.2 Cenário II - Grid

O cenário *II* é compostos por um Grid 4*x*4 com ruas de 2 quilômetros. O primeiro acidente possui aproximadamente 156 veículos o segundo possui aproximadamente 172 veículos. Os veículos que estão nas ruas dos acidentes ficam parados formando duas filas. Os outros veículos da simulação, que não passam por essas ruas, continuam em movimento normalmente. No total da simulação existe aproximadamente 3000 veículos.

Os resultados no cenário *II* são semelhantes aos do cenário *I*, o ReG3M manteve sendo melhor que os outros algoritmos em todas as métricas avaliadas em mensagens de 10*Mb* e 5*Mb*. Novamente pelo fato dos veículos ficarem ocupados até o recebimento da mensagem completa, o que ocasiona mais colisões na rede quando não existe um controle centralizado para reenvio das mensagens. Em mensagens menores de, 100*Kb* e 10*Kb*, o tempo de ocupação dos veículos é menor, e mesmo com um número elevado de colisões na rede os outros algoritmos conseguem ser eficientes e em alguns casos ligeiramente superiores ao ReG3M. Os gráficos a seguir mostram os resultados obtidos no cenário *II* para os quatro tamanhos de mensagem.

Teste com mensagens de 10 Mb

Nos teste com mensagens de 10*Mb* podemos observar que o ReG3M obteve maior número, em média, de veículos que receberam as mensagens nos dois acidentes. Como podemos ver nas Figuras 6.21 e 6.22.

Figura 6.21 – Número de veículos alcançados - Acidente 1 (10 Mb)

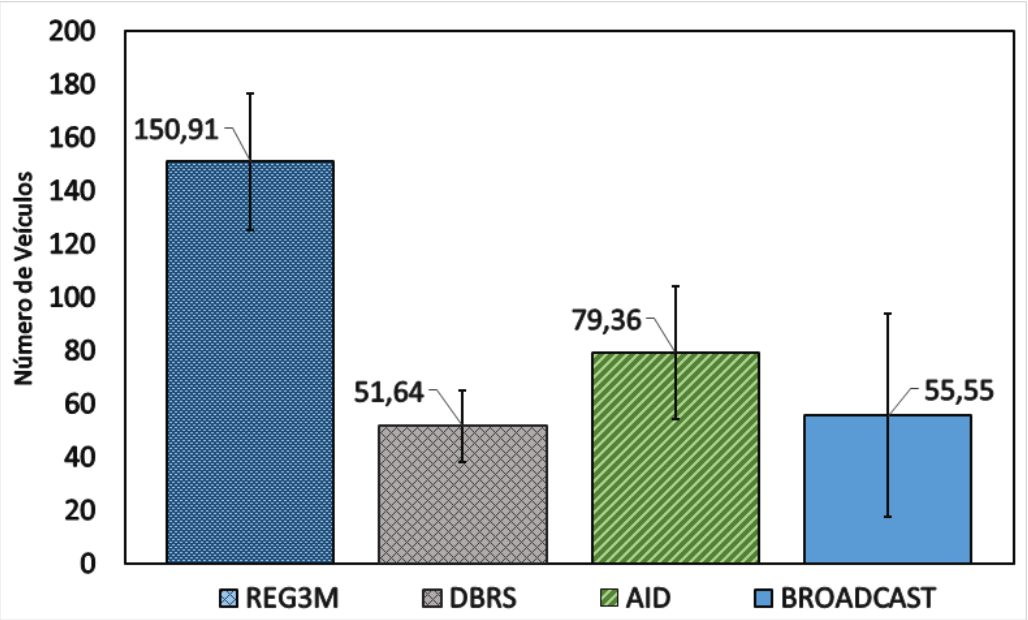
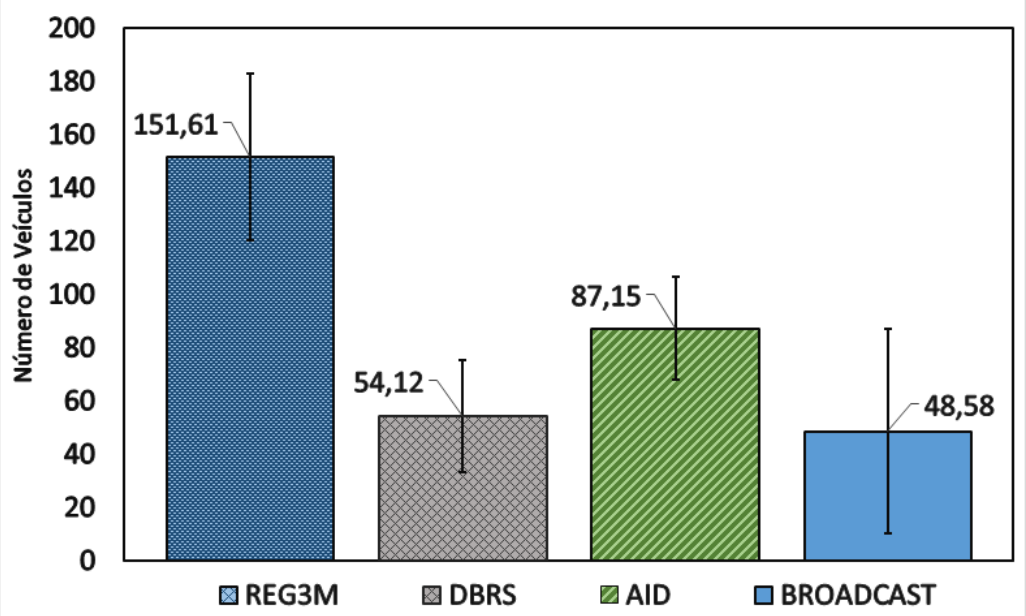


Figura 6.22 – Número de veículos alcançados - Acidente 2 (10 Mb)



O alcance considerando a distância da origem da mensagem é proporcional ao número de veículos, com isso o ReG3M também obteve resultados melhores que os outros algoritmos nos dois acidentes, como mostrado nas Figuras 6.23 e 6.24.

Figura 6.23 – Distância alcançada - Acidente 1 (10 Mb)

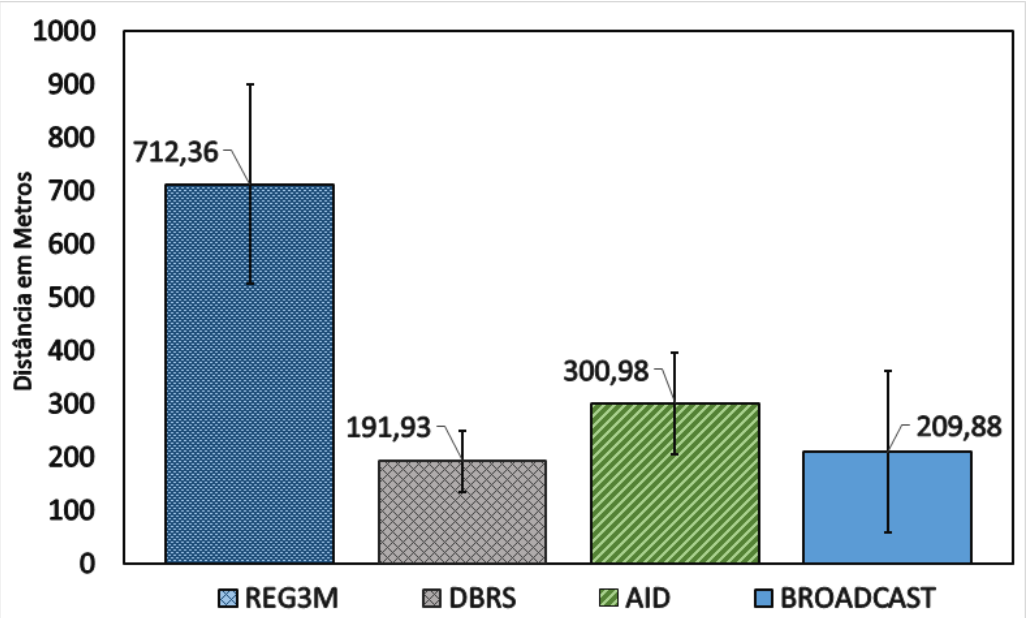
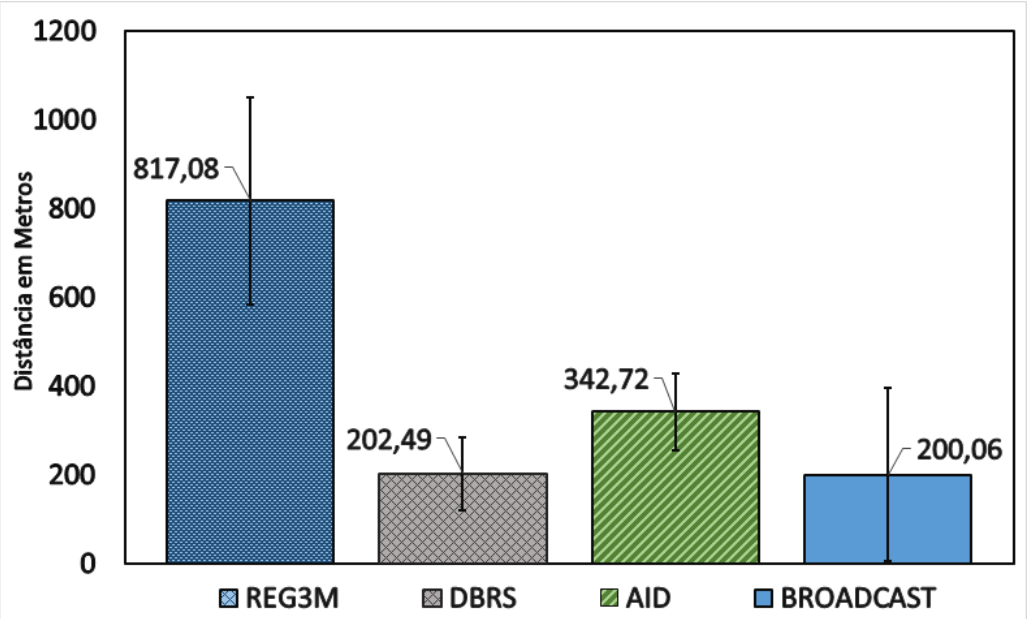


Figura 6.24 – Distância alcançada - Acidente 2 (10 Mb)



A relação entre o número de veículos alcançados e o número de encaminhadores das mensagens do ReG3M foi melhor, nos dois acidentes, que os resultados do AID e DBRS que também possuem controle de encaminhadores, conforme mostra as Figuras 6.25 e 6.26.

Figura 6.25 – Relação Encaminhadores x Veículos alcançados - Acidente 1 (10 Mb)

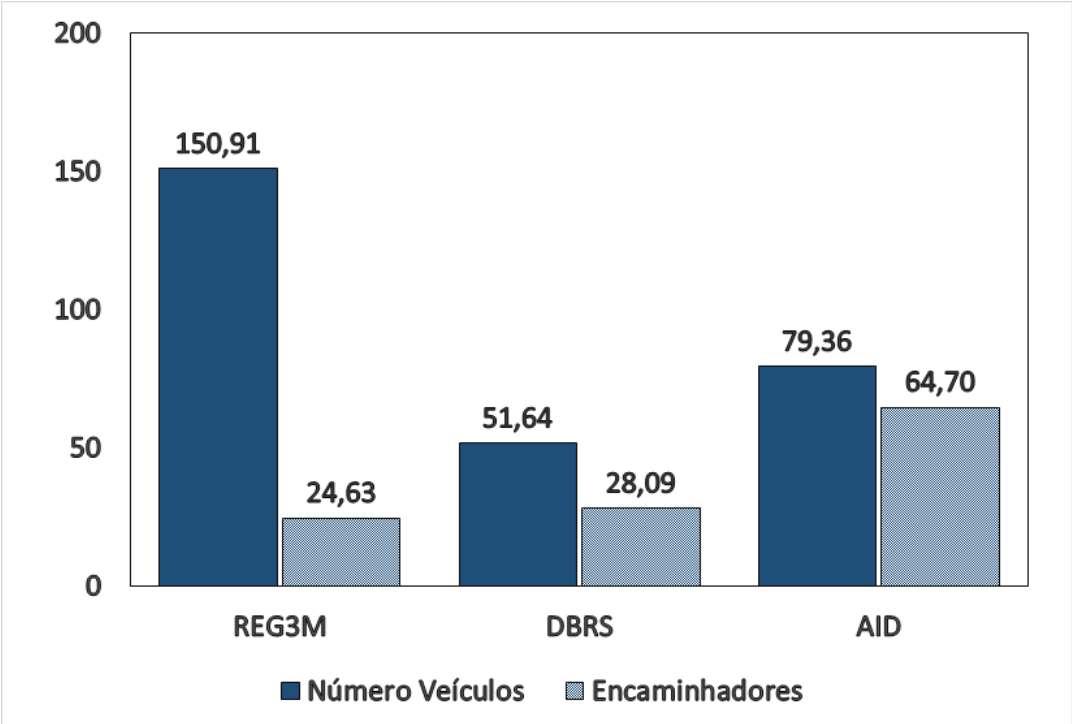
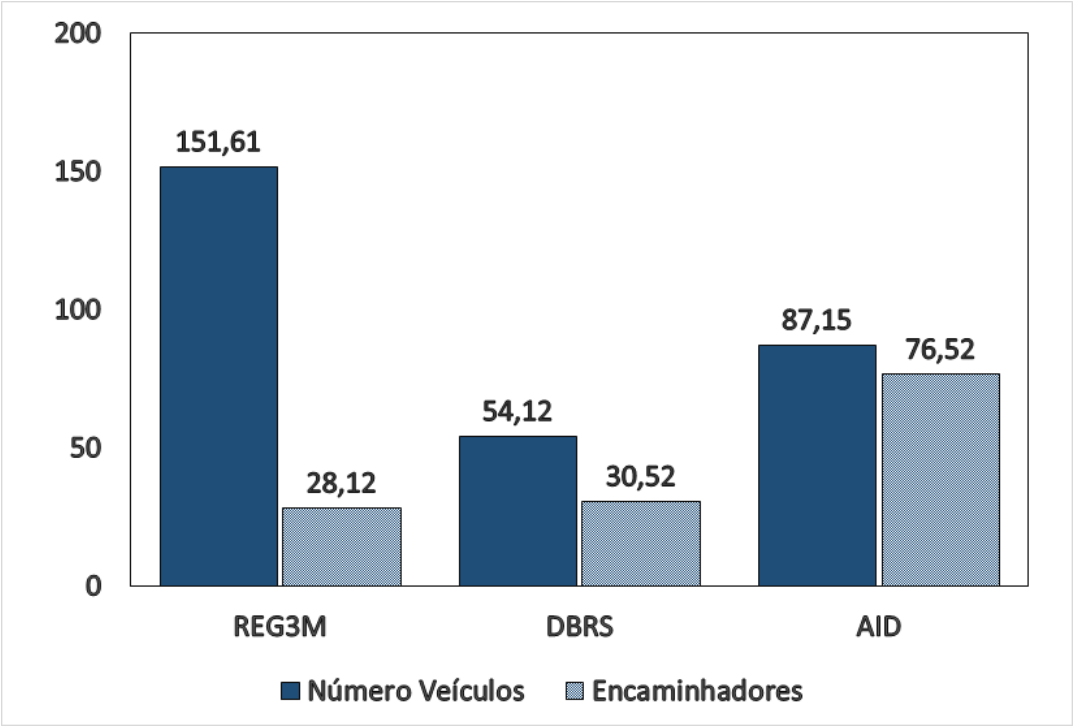
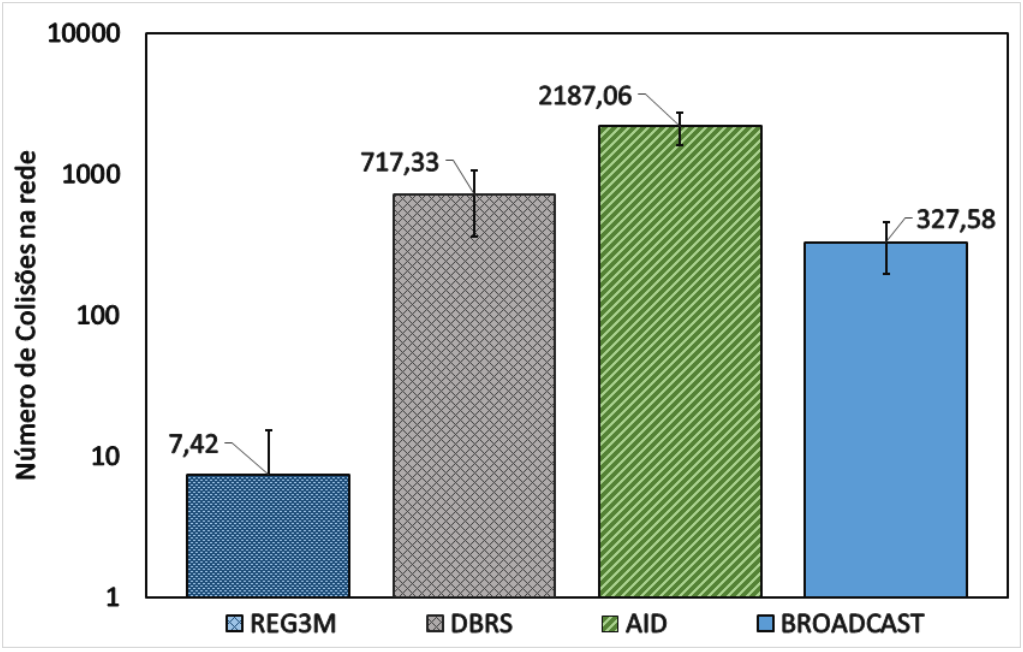


Figura 6.26 – Relação Encaminhadores x Veículos alcançados - Acidente 2 (10 Mb)



Os resultados do número de colisões na rede foram obtidos de todo o cenário, eles representam o total de colisões na rede levando em consideração os dois acidentes e os veículos que não estão envolvidos. A Figura 6.27 mostra que o ReG3M teve um número consideravelmente menor de colisões em seus experimentos.

Figura 6.27 – Colisões na Rede (10 Mb)



Teste com mensagens de 5 Mb

Nos teste com mensagens de 5*Mb* o ReG3M continua com o maior número de veículos alcançados, nos dois acidentes, mas esta praticamente empatado com o AID no acidente 1. No acidente 2 o ReG3M teve 74% mais veículos alcançados, como podemos ver nas Figuras 6.28 e 6.29.

Figura 6.28 – Número de veículos alcançados - Acidente 1 (5 Mb)

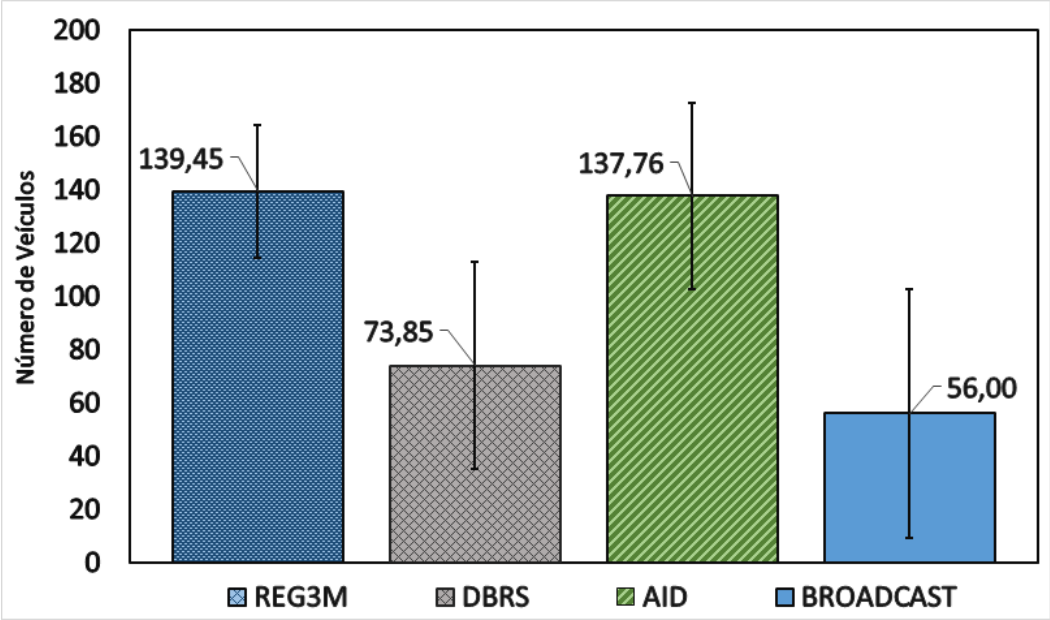
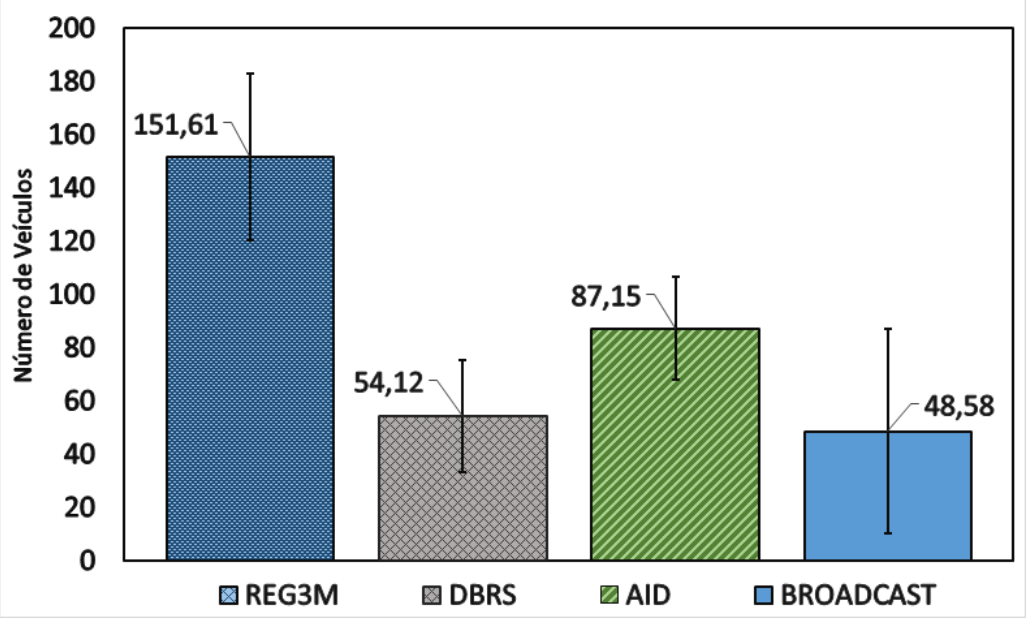


Figura 6.29 – Número de veículos alcançados - Acidente 2 (5 Mb)



A distância alcançada do ReG3M também obteve resultados melhores que os outros algoritmos nos dois acidentes. Ficando praticamente empatado com AID no acidente 1 e sendo 168,46 metros superior no acidente 2, como mostrado nas Figuras 6.30 e 6.31.

Figura 6.30 – Distância alcançada - Acidente 1 (5 Mb)

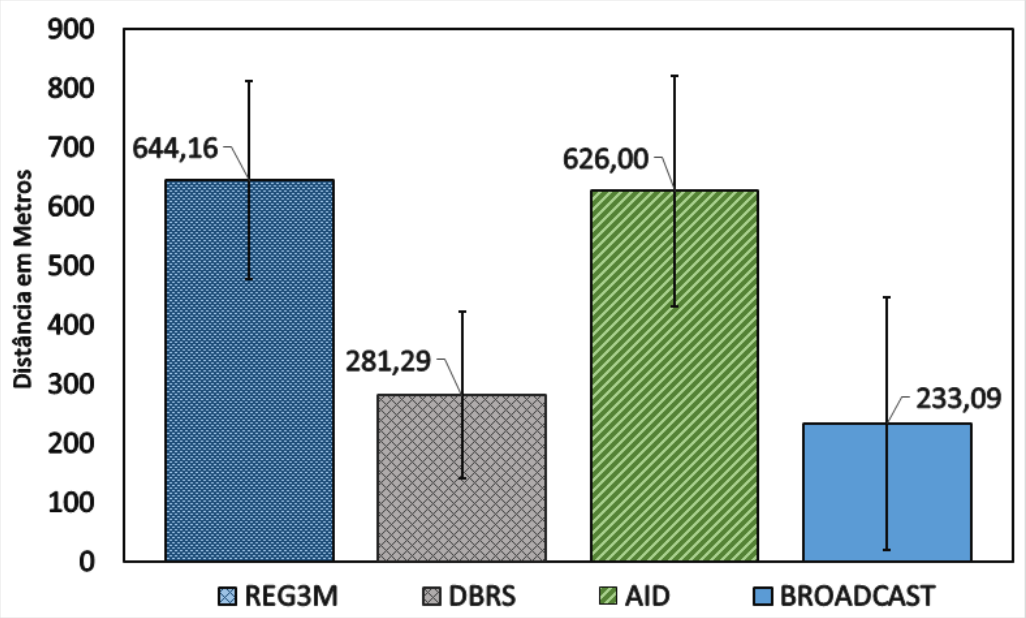
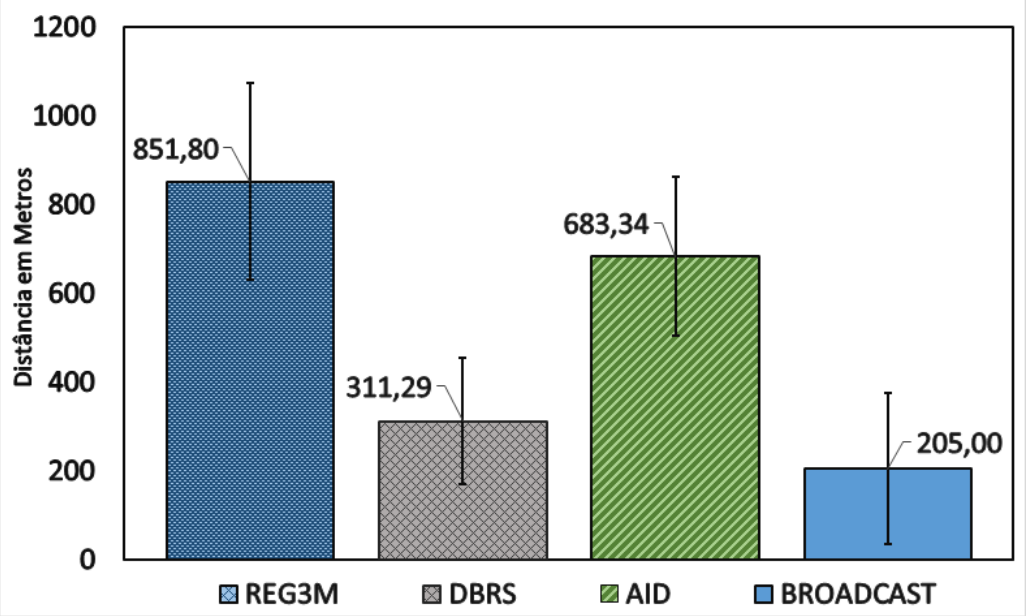


Figura 6.31 – Distância alcançada - Acidente 2 (5 Mb)



A relação entre o número de veículos e encaminhadores também teve valores melhores no ReG3M do que nos outros dois algoritmos. As Figuras 6.32 e 6.33 mostram que o ReG3M continua precisando de menos encaminhadores para alcançar mais veículos.

Figura 6.32 – Relação Encaminhadores x Veículos alcançados - Acidente 1 (5 Mb)

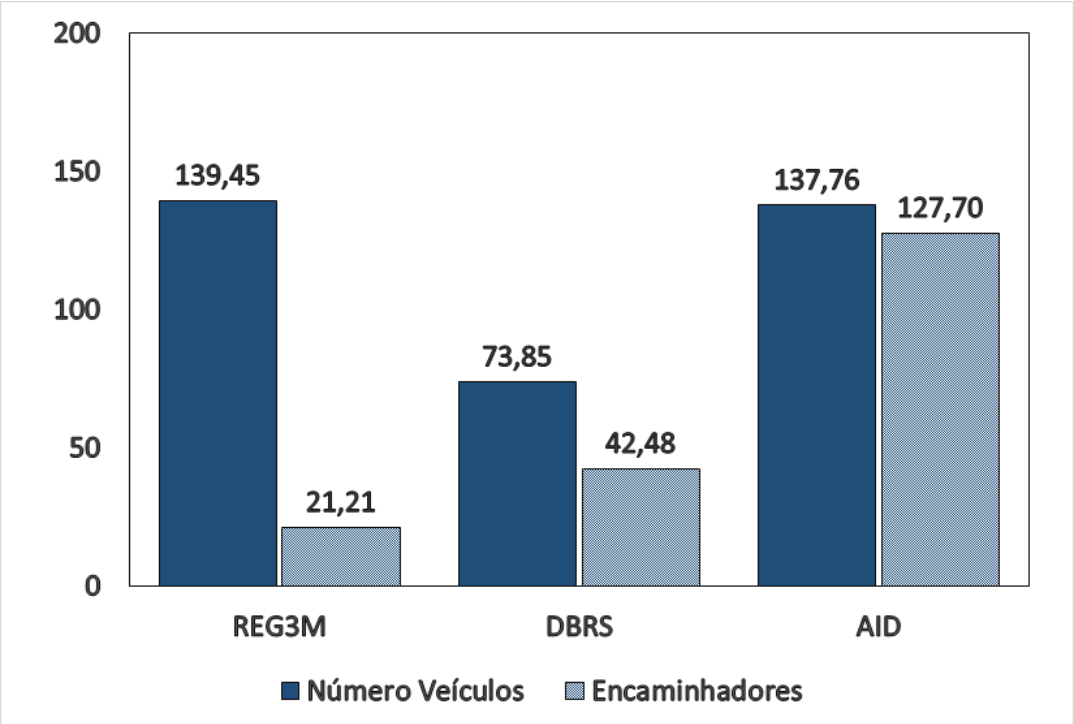
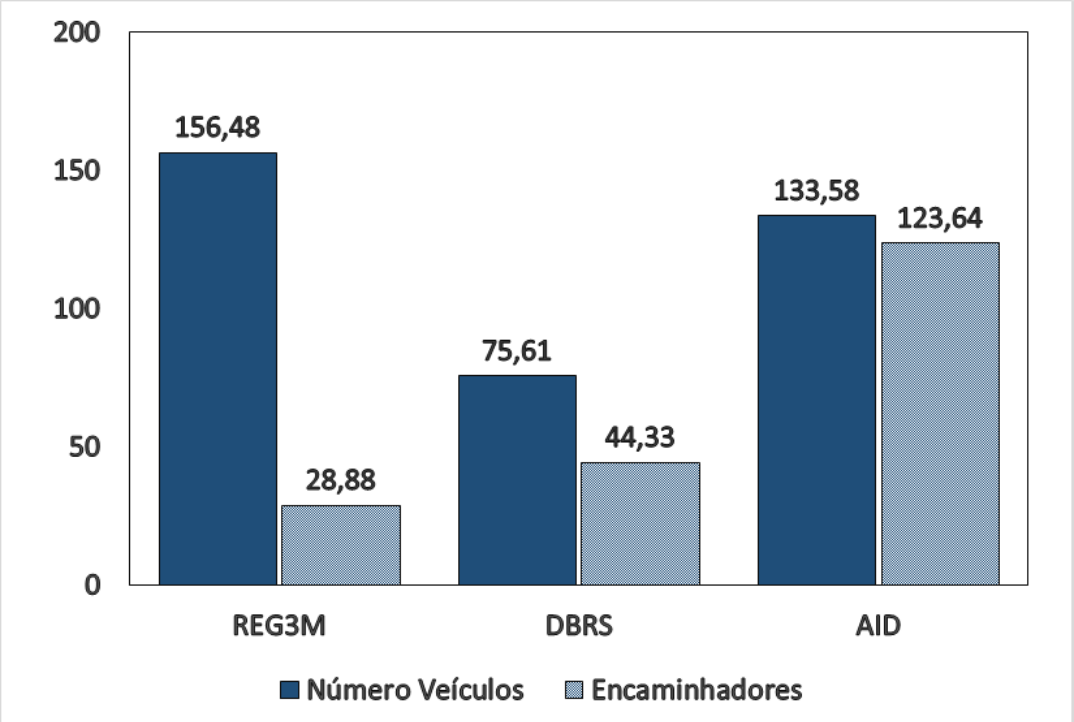
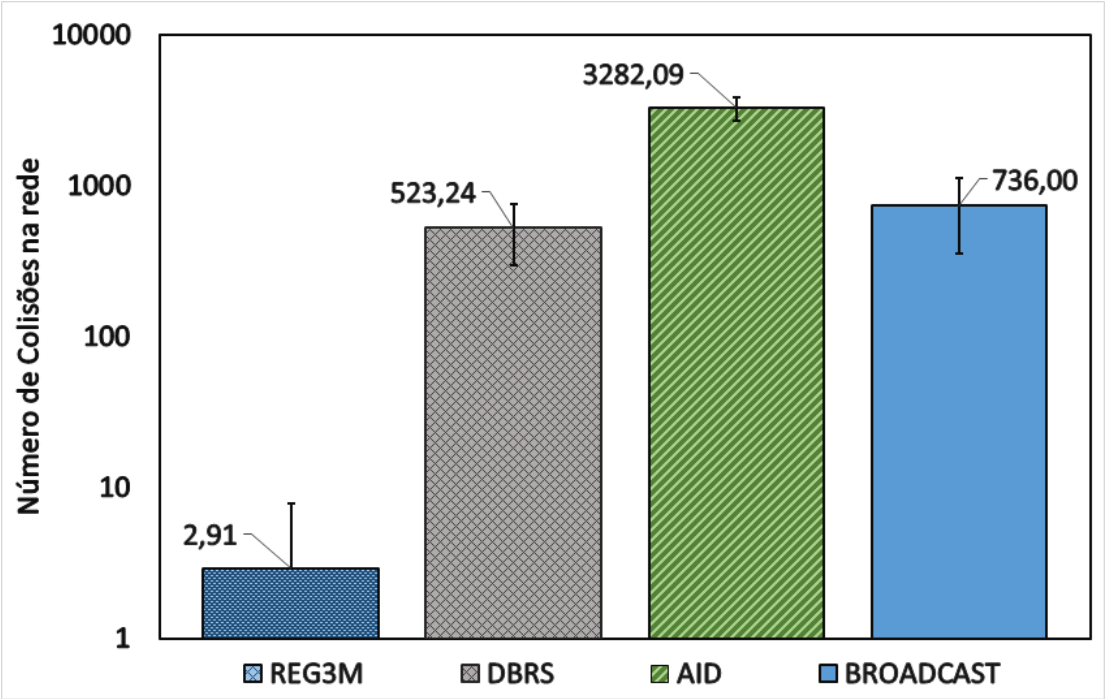


Figura 6.33 – Relação Encaminhadores x Veículos alcançados - Acidente 2 (5 Mb)



Os resultados do número de colisões mostram que o ReG3M consegue manter um número baixo de colisões na rede, conforme mostra a Figura 6.34

Figura 6.34 – Colisões na Rede (5 Mb)



Teste com mensagens de 100 Kb

Nos teste com mensagens de 100*Kb* o ReG3M foi ligeiramente superado pelo *Broadcast* na média de veículos alcançados no acidente 1. No acidente 2 o ReG3M continua sendo melhor na média, tendo alcançado 8% mais veículos. As Figuras 6.35 e 6.36 mostram os resultados nos dois acidentes.

Figura 6.35 – Número de veículos alcançados - Acidente 1 (100 Kb)

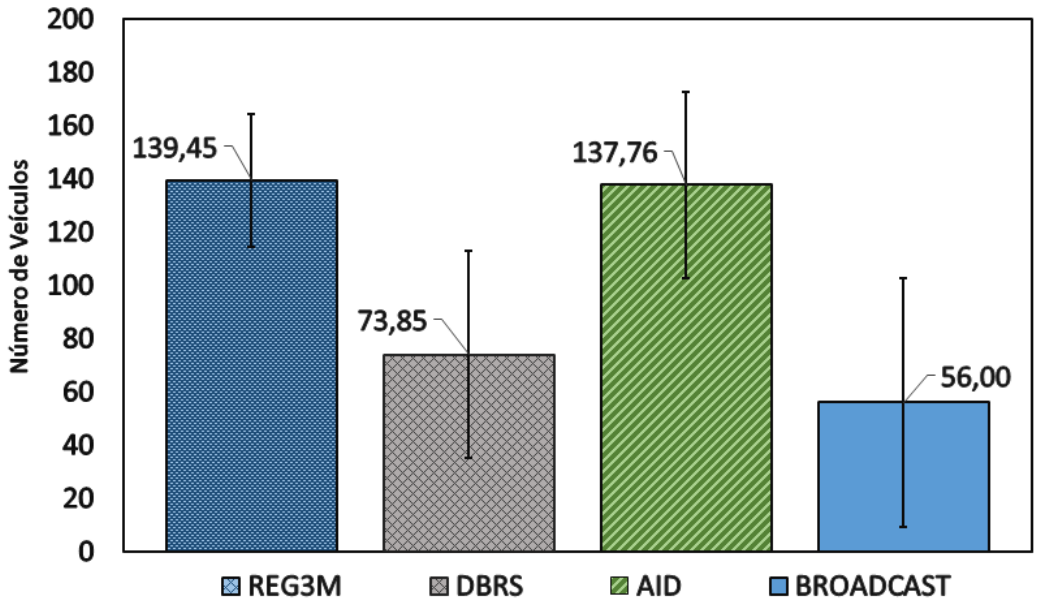
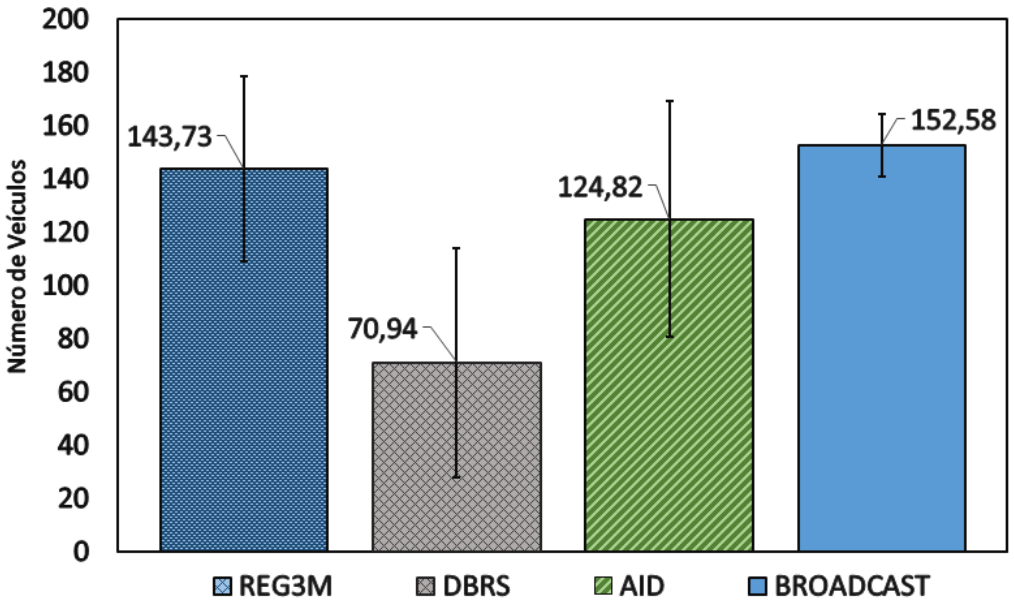


Figura 6.36 – Número de veículos alcançados - Acidente 2 (100 Kb)



A distância alcançada é proporcional a quantidade de veículos que receberam a mensagem, sendo assim o ReG3M teve resultado pior na média se comparado ao *Broadcast* no acidente 1 porém conseguiu ser melhor que ele no acidente 2. As Figuras 6.37 e 6.38 mostram os resultados nos dois acidentes.

Figura 6.37 – Distância alcançada - Acidente 1 (100 Kb)

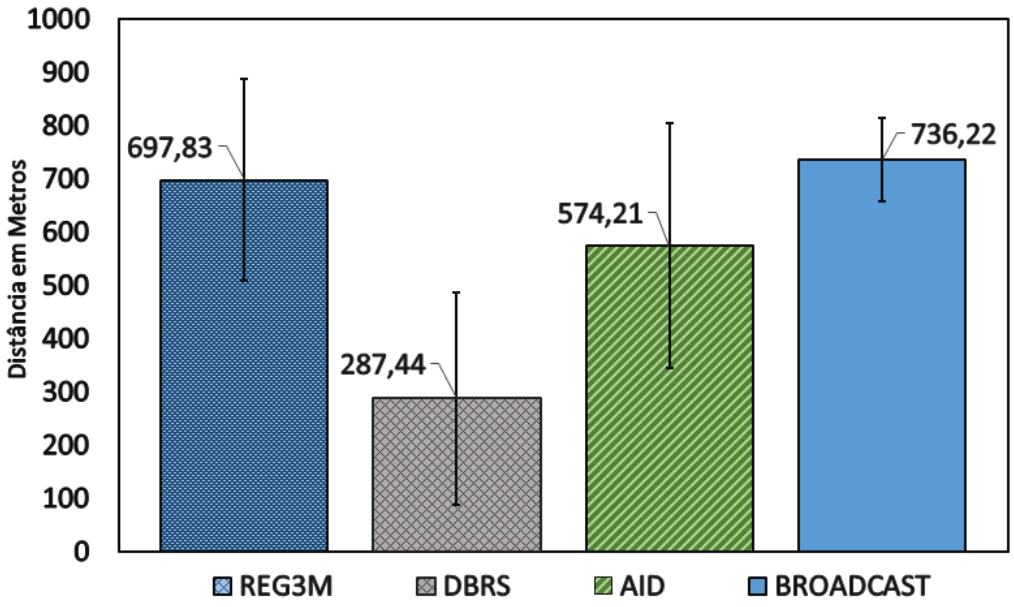
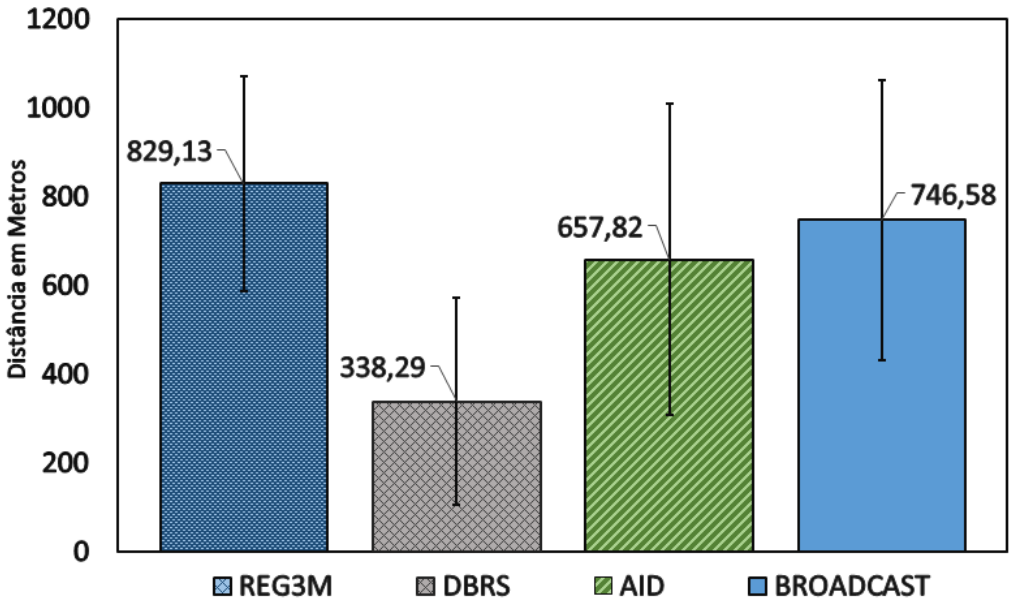


Figura 6.38 – Distância alcançada - Acidente 2 (100 Kb)



A relação entre o número de veículos e encaminhadores o ReG3M continua sendo melhor do que os outros dois veículos. As Figuras 6.39 e 6.40 mostram os resultados.

Figura 6.39 – Relação Encaminhadores x Veículos alcançados - Acidente 1 (100 Kb)

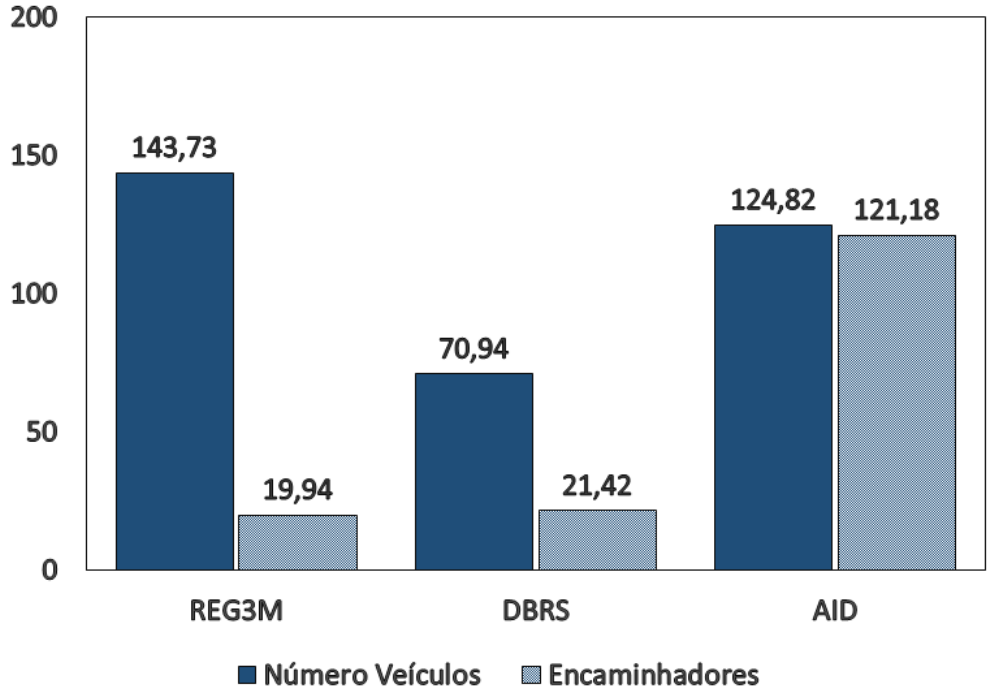
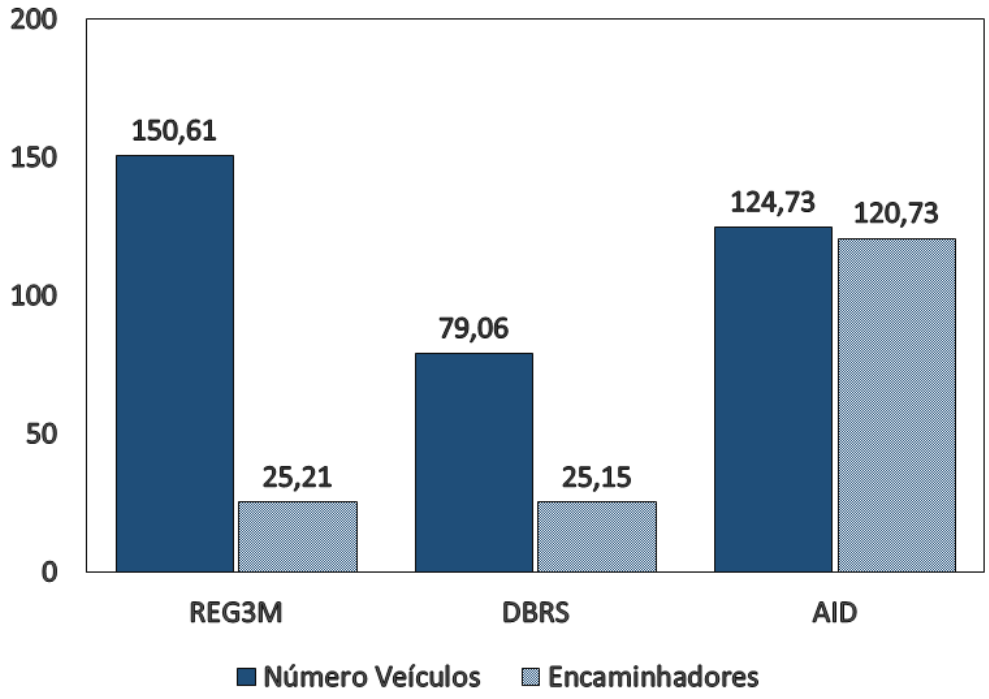
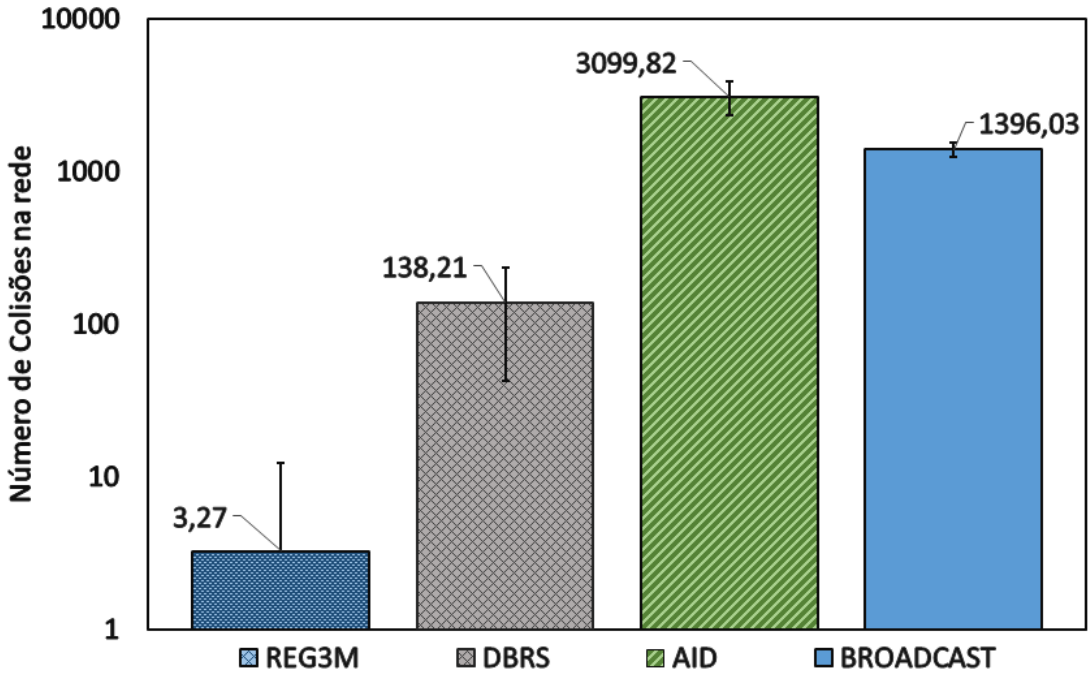


Figura 6.40 – Relação Encaminhadores x Veículos alcançados - Acidente 2 (100 Kb)



Em relação ao número de colisões na rede, o ReG3M ainda tem números muito menores que os outros algoritmos conforme mostrado na Figura 6.41

Figura 6.41 – Colisões na Rede (100 Kb)



Teste com mensagens de 10 Kb

Nos teste com mensagens de 10*Kb* o ReG3M foi superado pelo *Broadcast* e pelo AID, os dois algoritmos alcançaram todos os veículos que estavam parados nos dois acidentes. Levando em consideração o desvio padrão, o ReG3M e o DBRS, em alguns casos conseguiram enviar para todos os que estavam parados e alguns que chegaram um pouco depois e estavam desacelerando, com isso pode ser considerado um empate entre os 4 algoritmos. As Figuras 6.42 e 6.43 mostram os resultados nos dois acidentes.

Figura 6.42 – Número de veículos alcançados - Acidente 1 (10 Kb)

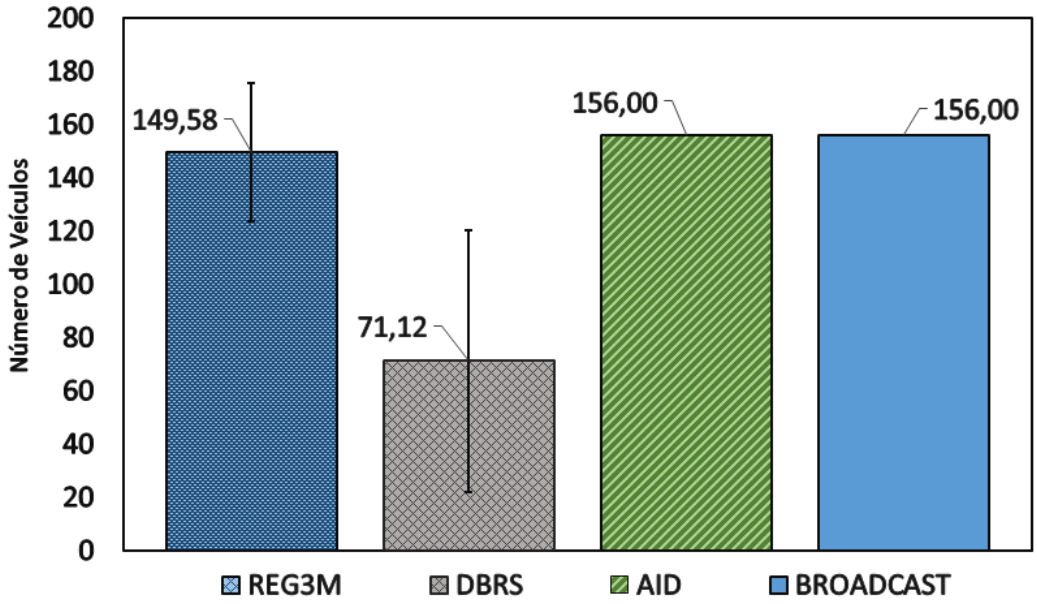
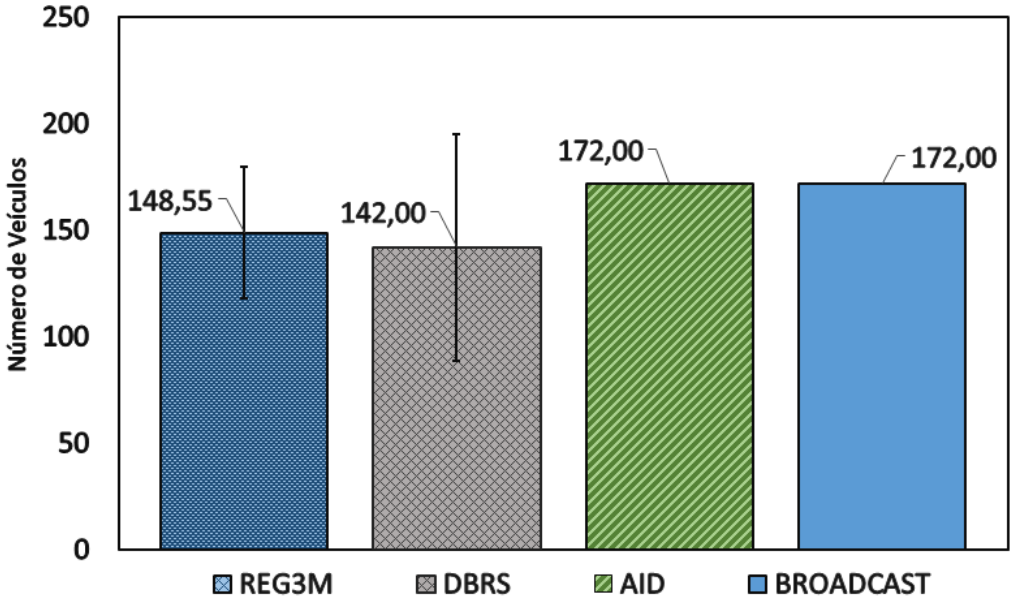


Figura 6.43 – Número de veículos alcançados - Acidente 2 (10 Kb)



A mesma coisa que aconteceu com o número de veículos acontece com a distância. O ReG3M foi superado pelo AID e *Broadcast* mas em alguns casos atingiu uma distância maior por conseguir enviar para veículos que estavam chegando ao congestionamento. As Figuras 6.44 e 6.45 mostram os resultados nos dois acidentes.

Figura 6.44 – Distância alcançada - Acidente 1 (10 Kb)

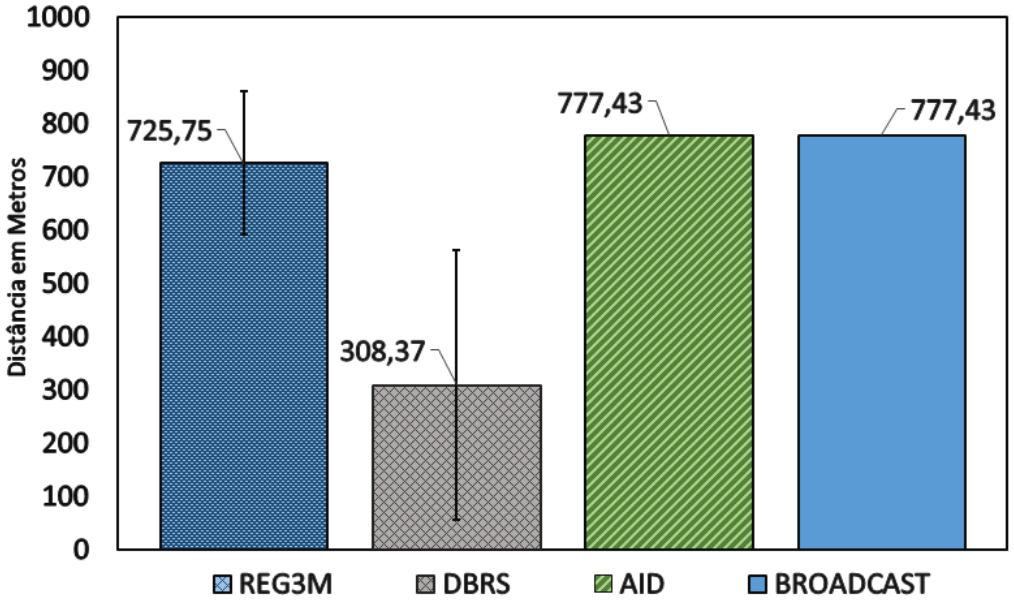
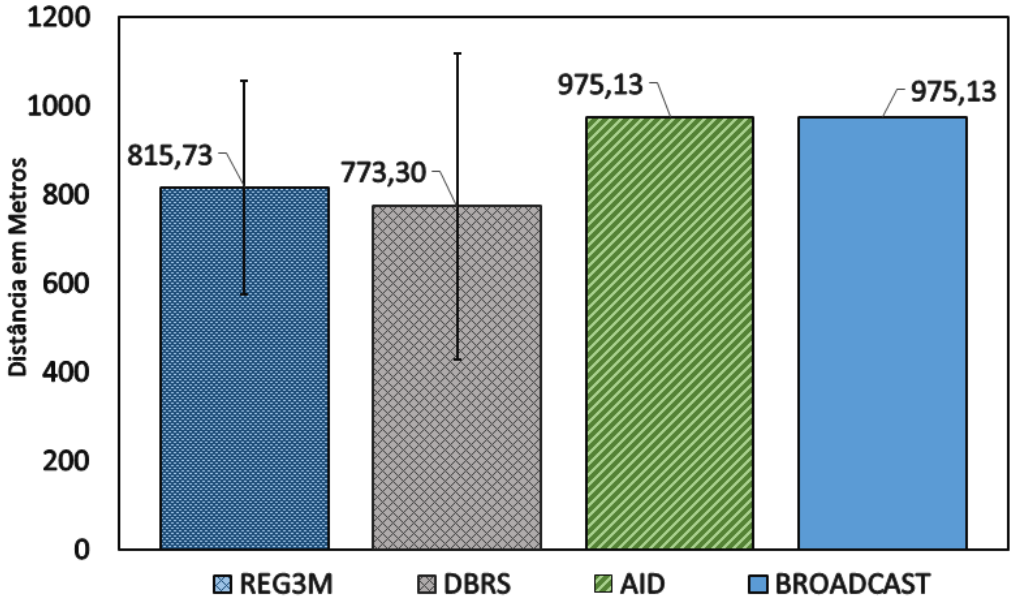


Figura 6.45 – Distância alcançada - Acidente 2 (10 Kb)



A relação entre o número de veículos e encaminhadores o ReG3M mantém um número mais baixo que os outros algoritmos. As Figuras 6.46 e 6.47 mostram os resultados.

Figura 6.46 – Relação Encaminhadores x Veículos alcançados - Acidente 1 (10 Kb)

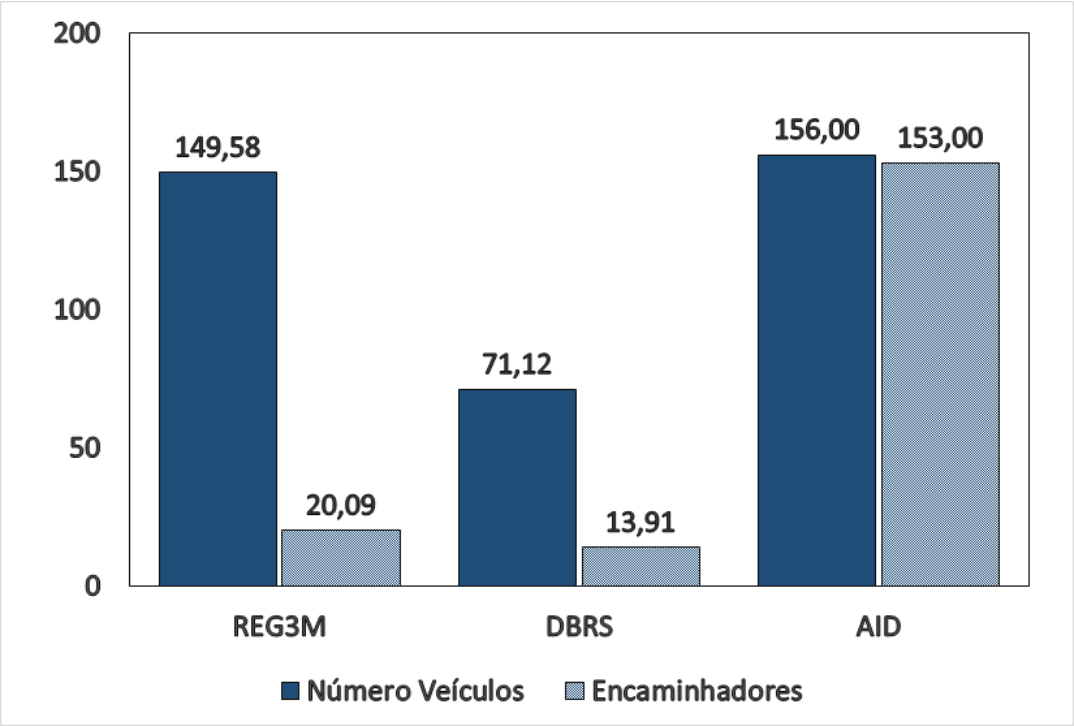
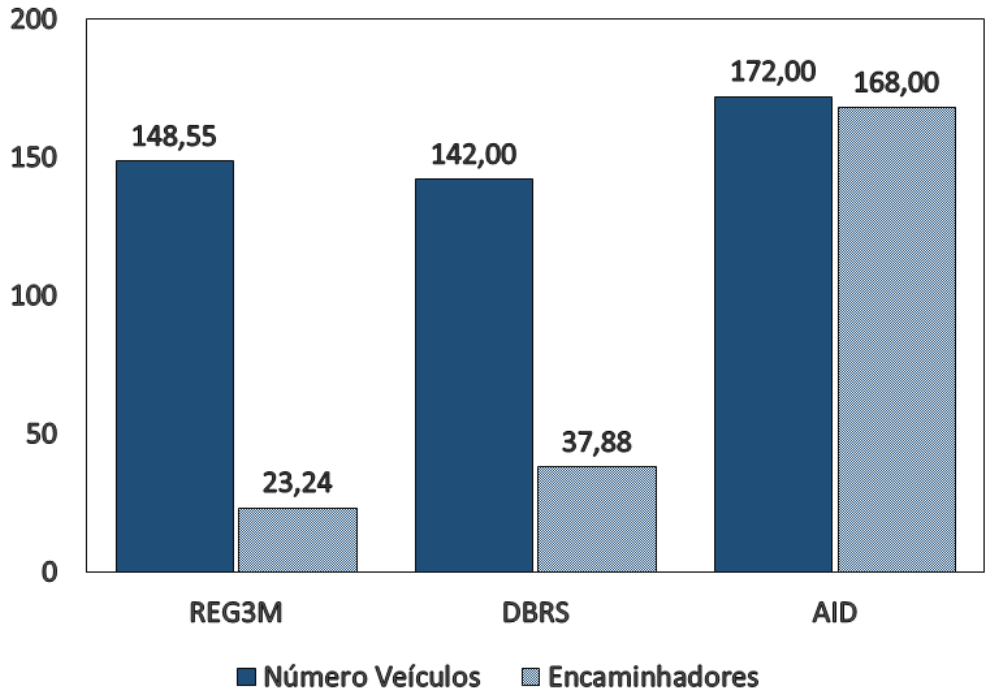
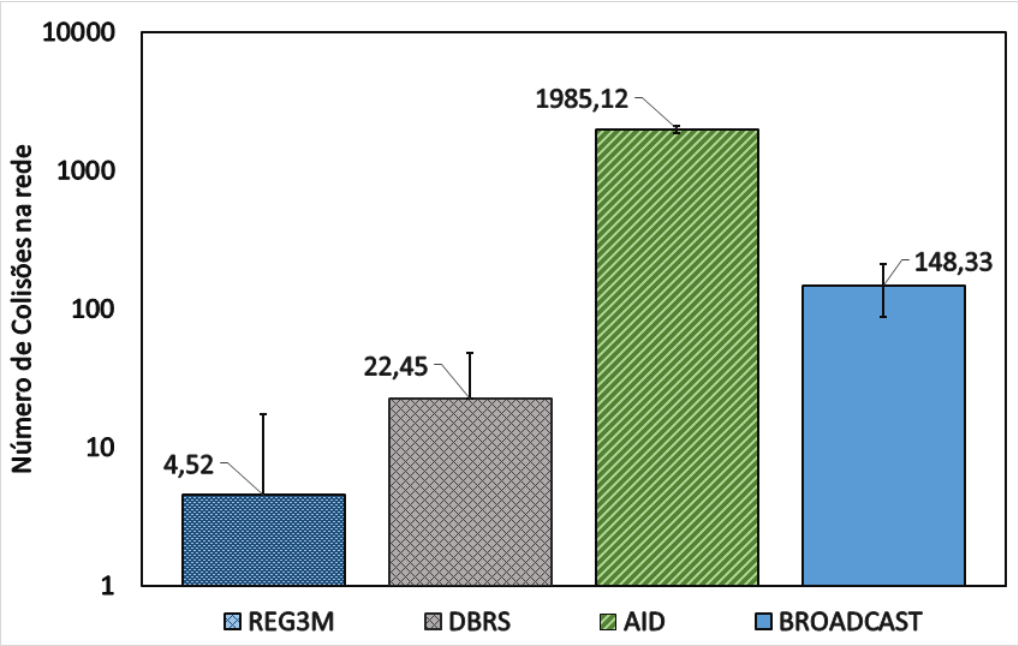


Figura 6.47 – Relação Encaminhadores x Veículos alcançados - Acidente 2 (10 Kb)



Em relação ao número de colisões na rede, o ReG3M se manteve superior em todos os teste, tendo números muito menores que os outros algoritmos. A Figura 6.48 mostra os resultados.

Figura 6.48 – Colisões na Rede (10 Kb)



6.2.3 Cenário III

O cenário *III* foi baseado no cenário *I* da subseção 6.2.1, porém neste caso serão avaliados o tempo gasto para as mensagens de 10*Mb*, 5*Mb*, 100*Kb* e 10*Kb* alcançarem as distância de 100, 200, 300, 400, 500, 600, 700 e 800 metros. Além disso foi avaliado o percentual de vezes que as mensagens alcançaram essas medidas dentro das 33 simulações realizadas para cada tamanho de mensagem, considerando 120s desde o envio da primeira mensagem.

No cenário *III* o ReG3M foi novamente superior aos outros algoritmos em mensagens de 10*Mb* e 5*Mb*, conseguindo percorrer todas as distâncias em menos tempo. Em mensagens de 100*Kb* ele foi superado no tempo gasto para percorrer as distâncias apenas pelo DBRS, já em mensagens de 10*Kb* o ReG3M foi superado por todos os algoritmos. O ReG3M consegue ter resultados melhores do que os outros algoritmos pois, tem um controle centralizado no reencaminhamento das mensagens, com isso necessita de menos tentativas de envio, o que garante eficiência na entrega das mensagens.

Teste com mensagem de 10 Mb

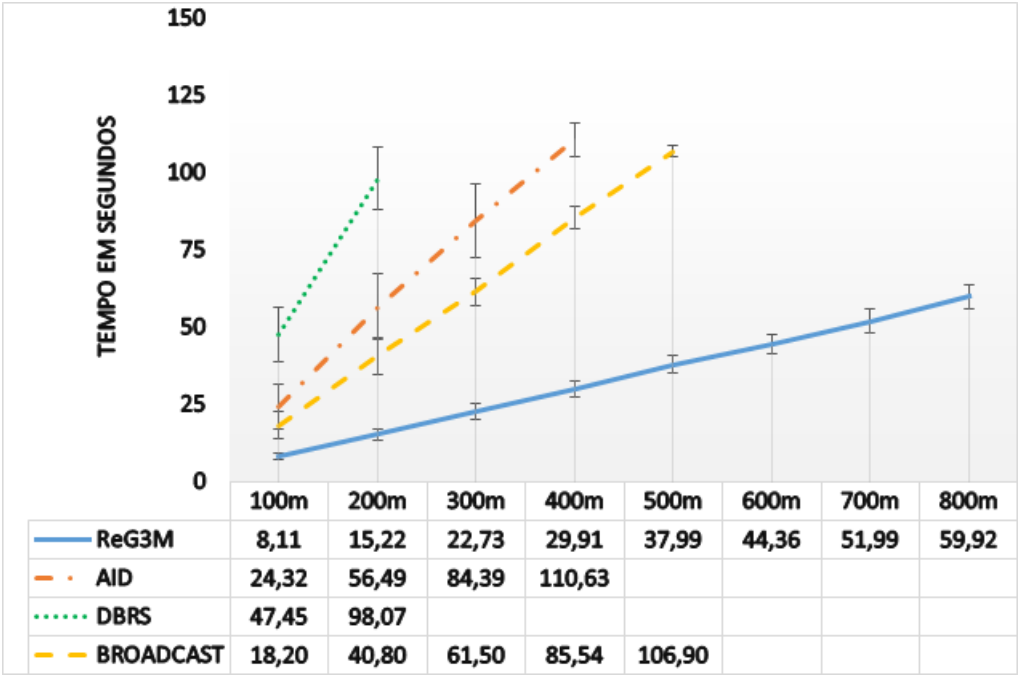
A tabela 6.5 mostra o percentual de vezes que cada um dos algoritmos alcançou cada distância. O ReG3M foi o que manteve o melhor percentual e o único que alcançou todas as distâncias.

Tabela 6.5 – Percentual de alcance (10 Mb)

	ReG3M	AID	DBRS	Broadcast
100m	100%	84.85%	93.94%	78.79%
200m	96.97%	75.76%	69.70%	36.36%
300m	90.91%	72.73%	0%	15.15%
400m	90.91%	39.39%	0%	15.15%
500m	90.91%	0%	0%	0%
600m	87.88%	0%	0%	0%
700m	84.84%	0%	0%	0%
800m	84.84%	0%	0%	0%

Além de alcançar todas as distância durante os teste, o ReG3M também foi o mais rápido em propagar a mensagem entre os veículos, conforme mostra a Figura 6.49.

Figura 6.49 – Propagação da mensagem (10 Mb)



Teste com mensagem de 5 Mb

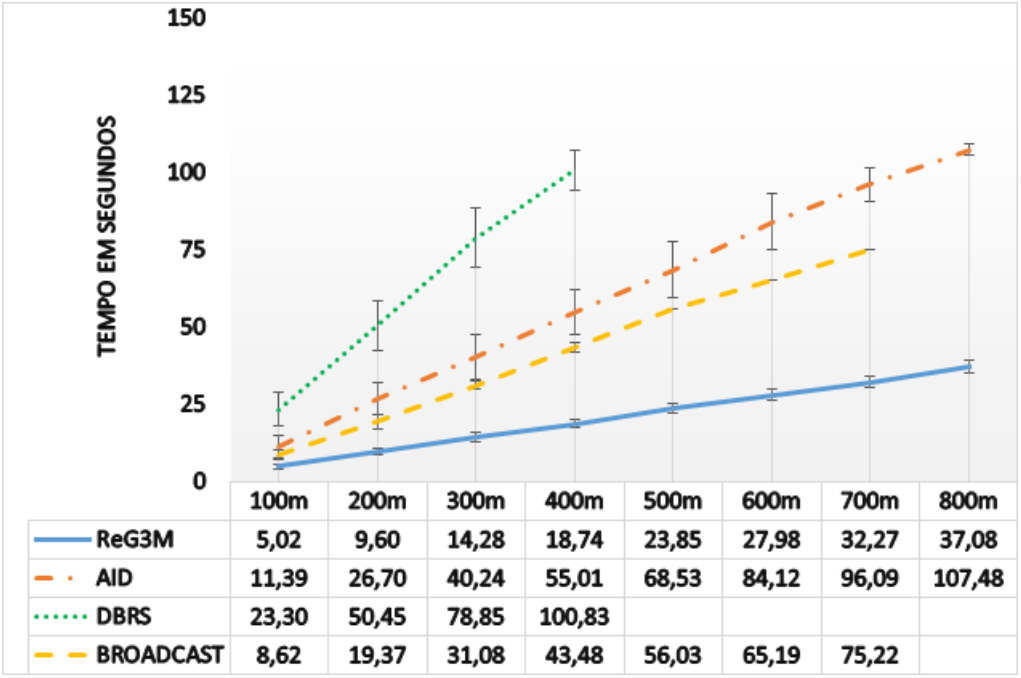
A tabela 6.6 mostra o percentual de vezes que cada um dos algoritmos alcançou cada distância nos teste com mensagens de 5Mb, o ReG3M manteve o melhor percentual de vezes que alcançou cada distância, mas podemos observar uma melhora nos outros algoritmos em relação ao teste anterior.

Tabela 6.6 – Percentual de alcance (5 Mb)

	ReG3M	AID	DBRS	Broadcast
100m	100.00%	78.79%	96.97%	63.64%
200m	96.97%	69.70%	93.94%	36.36%
300m	90.91%	60.61%	81.82%	18.18%
400m	87.88%	54.55%	48.48%	6.06%
500m	87.88%	42.42%	0.00%	3.03%
600m	81.82%	39.39%	0.00%	3.03%
700m	81.82%	27.27%	0.00%	3.03%
800m	78.79%	6.06%	0.00%	0.00%

Considerando a velocidade de propagação da mensagem, o ReG3M mostrou-se novamente melhor que os outros algoritmos, sendo o mais rápido entre os algoritmos em todas as distância. A Figura 6.50 mostra os resultados.

Figura 6.50 – Propagação da mensagem (5 Mb)



Teste com mensagem de 100 Kb

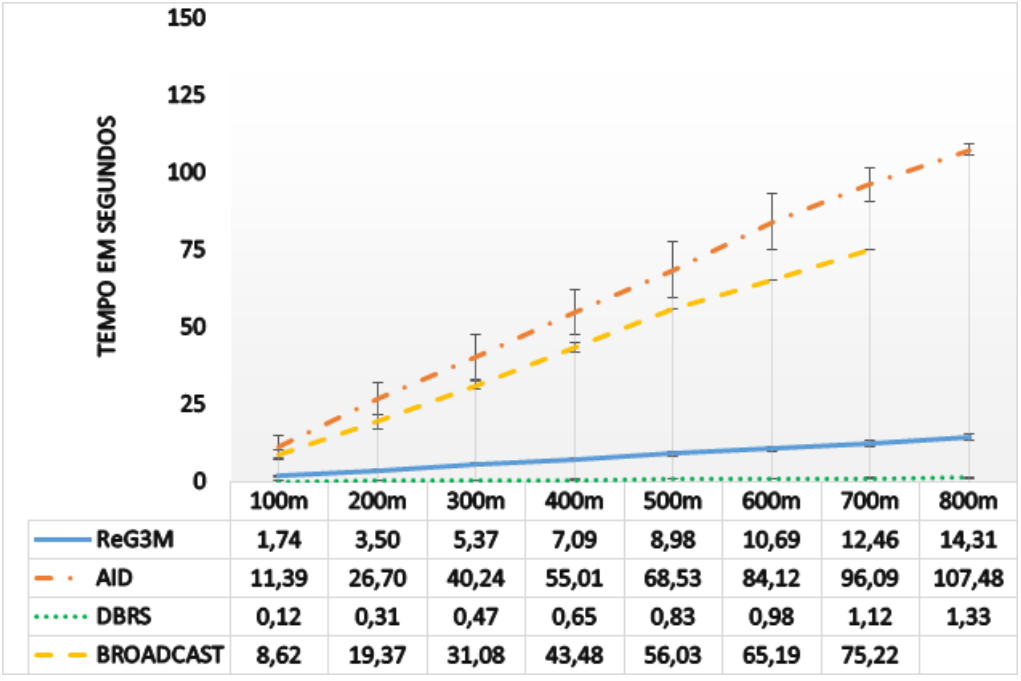
A tabela 6.7 mostra o percentual de vezes que cada um dos algoritmos alcançou cada distância nos teste com mensagens de 100Kb, o ReG3M ainda manteve o melhor percentual em todas as distâncias, porém os outros algoritmos melhoraram seus percentuais e alcance, em relação aos testes anteriores.

Tabela 6.7 – Percentual de alcance (100 Kb)

	ReG3M	AID	DBRS	Broadcast
100*m*	100.00%	78.79%	93.94%	63.64%
200*m*	100.00%	69.70%	72.73%	36.36%
300*m*	96.97%	60.61%	57.58%	18.18%
400*m*	93.94%	54.55%	48.48%	6.06%
500*m*	93.94%	42.42%	33.33%	3.03%
600*m*	93.94%	39.39%	30.30%	3.03%
700*m*	90.91%	27.27%	18.18%	3.03%
800*m*	87.88%	6.06%	15.15%	0.00%

Para mensagens de 100*Kb* o ReG3M foi mais lento em propagação apenas para o algoritmo DBRS. Porém levando em consideração aos percentuais apresentados na Tabela 6.7, o ReG3M, apesar de mais lento, consegue por mais vezes levar a mensagem mais longe que o DBRS, conforme mostrado na Figura 6.51.

Figura 6.51 – Propagação da mensagem (100 Kb)



Teste com mensagem de 10 Kb

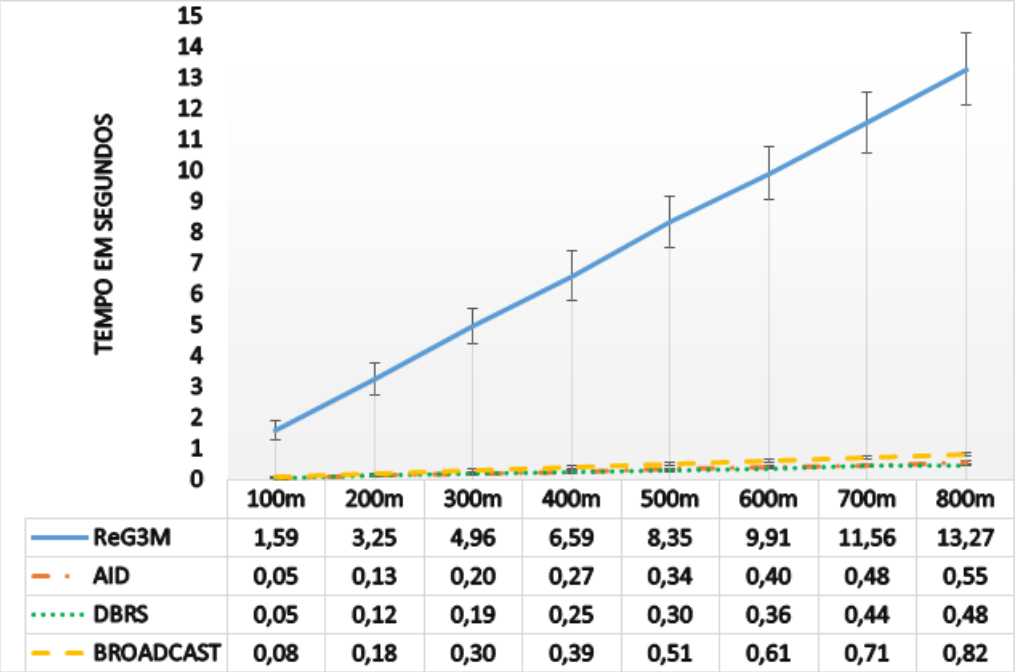
A tabela 6.8 mostra o percentual de vezes que cada um dos algoritmos alcançou cada distância nos teste com mensagens de 10*Kb*. Para testes com esse tamanho de mensagens o ReG3M, apesar de manter um percentual alto, foi pior que os algoritmos AID e *Broadcast* que obtiveram 100% de eficiência na entrega das mensagens.

Tabela 6.8 – Percentual de alcance (10 Kb)

	ReG3M	**AID**	**DBRS**	**Broadcast**
100*m*	96.97%	100.00%	96.97%	100.00%
200*m*	96.97%	100.00%	72.73%	100.00%
300*m*	93.94%	100.00%	51.52%	100.00%
400*m*	93.94%	100.00%	39.39%	100.00%
500*m*	87.88%	100.00%	24.24%	100.00%
600*m*	87.88%	100.00%	18.18%	100.00%
700*m*	90.91%	100.00%	9.09%	100.00%
800*m*	87.88%	100.00%	9.09%	100.00%

Em relação a velocidade de propagação, o ReG3M foi mais lento que todos os algoritmos, sendo consideravelmente mais lento que os demais. Isso ocorre pelo fato de ter que esperar a resposta das distância que receberam a mensagem, para poder indicar seu encaminhador. Como mostra a Figura 6.52.

Figura 6.52 – Propagação da mensagem (10 Kb)



	100m	200m	300m	400m	500m	600m	700m	800m
ReG3M	1,59	3,25	4,96	6,59	8,35	9,91	11,56	13,27
AID	0,05	0,13	0,20	0,27	0,34	0,40	0,48	0,55
DBRS	0,05	0,12	0,19	0,25	0,30	0,36	0,44	0,48
BROADCAST	0,08	0,18	0,30	0,39	0,51	0,61	0,71	0,82

7 CONCLUSÃO E TRABALHOS FUTUROS

Para a transmissão da mensagem de vídeos, foi apresentado o ReG3M, um algoritmo que otimiza a transmissão de arquivos grandes, reduzindo colisões e retransmissões nas VANETs. O ReG3M obteve resultados melhores que os algoritmos AID, DBRS e envio por *Broadcast*, para mensagens de $10Mb$ e $5Mb$ nos três cenários apresentados em relação à número de veículos alcançados, distância percorrida pela mensagem da origem até o último veículo que recebeu e velocidade de propagação da mensagem.

Para mensagens de $100Kb$ o ReG3M ainda conseguiu resultados, em média, melhor que os outros algoritmos. No cenário *II* o ReG3M ficou praticamente empatado com a transmissão em *broadcast* no acidente *II* se levarmos em consideração o desvio padrão nos resultados dos testes. Com esse tamanho de mensagem o ReG3M foi superado em velocidade de propagação apenas para o algoritmos DBRS.

Para mensagens de $10Kb$, em relação a media obtida nos testes de número de veículos alcançados e distância percorrida pela mensagem, o ReG3M foi melhor que os outros algoritmos no Cenário *I*. No Cenário *II* o ReG3M foi superado pelos algoritmos AID e *Broadcast*, e ficou um pouco a frente do algoritmo DBRS. Porém, como a diferença foi pequena, pode ser considerado um empate entre os quatro algoritmos. E em relação a velocidade de propagação apresentados no Cenário *III* o ReG3M foi o mais lento entre todos os algoritmos. Em consideração ao número de colisões na rede, o ReG3M foi o que menos apresentou colisões em todos os testes dos três cenários. Na comparação entre os algoritmos que apresentam escolhas de encaminhadores da mensagem, AID, DBRS e ReG3M. O ReG3M precisou de menos encaminhadores em todos os testes nos três cenários apresentados. Com os resultados apresentados nas transmissões podemos concluir que o ReG3M é uma opção viável para transmissão de mensagens maiores, como por exemplo os vídeos gerados pelo processo de homografia que foi apresentado neste trabalho.

O uso de homografia de vídeos em redes veiculares pode auxiliar os motoristas no trânsito, trazendo mais conforto e segurança em condições de congestionamento. Para a criação da mensagem de vídeo este trabalho apresentou o uso de homografia em VANET e mostrou uma comparação entre os descritores, em vídeos com diferentes resoluções. Os resultados mostraram que, quanto mais alta a resolução do vídeo, mais demorado é o processo para a junção das imagens. O descritor ORB foi o mais rápido em tempo de processamento e utilizou a menor quantidade de *Match* em suas comparações. A qualidade dos vídeos gerados foi muito pare-

cida em todos os descritores, os descritores BRISK e SIFT obtiveram melhores pontuação mas muito próximas dos outros descritores. O descritor BRISK foi o melhor na relação Tempo *x* Qualidade.

Como trabalhos futuros, pretende-se reduzir o tempo de processamento da homografia em vídeos. Pretende-se ainda investigar o uso de homografia em *streams* de vídeo e a escolha do melhor CODEC para uso em redes veiculares. Para tanto é necessário a adaptação do `ReG3M` para sua disseminação.

REFERÊNCIAS

AHMED, I. T.; CHEN, S. D.; HAMMAD, B. T. Recent approaches on no- reference image quality assessment for contrast distortion images with multiscale geometric analysis transforms: A survey. **Journal of Theoretical and Applied Information Technology**, Asian Research Publishing Network (ARPN), v. 95, n. 3, p. 561–569, 2 2017. ISSN 1992-8645.

AL-SULTAN, S. et al. A comprehensive survey on vehicular ad hoc network. **Journal of Network and Computer Applications**, v. 37, p. 380 – 392, 2014. ISSN 1084-8045. Disponível em: <http://www.sciencedirect.com/science/article/pii/S108480451300074X>.

BAKHOUYA, M.; GABER, J.; LORENZ, P. An adaptive approach for information dissemination in vehicular ad hoc networks. **Journal of Network and Computer Applications**, v. 34, n. 6, p. 1971 – 1978, 2011. ISSN 1084-8045. Control and Optimization over Wireless Networks. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1084804511001226>.

BAY, H. et al. Speeded-up robust features (surf). **Computer Vision and Image Understanding**, v. 110, n. 3, p. 346 – 359, 2008. ISSN 1077-3142. Similarity Matching in Computer Vision and Multimedia. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1077314207001555>.

BELAND, L.-P.; BRENT, D. A. Traffic and crime. **Journal of Public Economics**, v. 160, p. 96 – 116, 2018. ISSN 0047-2727. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0047272718300422>.

BRADSKI, D. G. R.; KAEHLER, A. **Learning Opencv, 1st Edition**. First. [S.l.]: O'Reilly Media, Inc., 2008. ISBN 9780596516130.

BRADSKI, G. The OpenCV Library. **Dr. Dobb's Journal of Software Tools**, 2000.

BROWN, M.; LOWE, D. G. Automatic panoramic image stitching using invariant features. **International Journal of Computer Vision**, Springer Nature, v. 74, n. 1, p. 59–73, dez. 2006. Disponível em: <https://doi.org/10.1007/s11263-006-0002-3>.

BROWN, M.; LOWE, D. G. Automatic panoramic image stitching using invariant features. **International Journal of Computer Vision**, v. 74, n. 1, p. 59–73, Aug 2007. Disponível em: <https://doi.org/10.1007/s11263-006-0002-3>.

CALONDER, M. et al. Brief: Binary robust independent elementary features. In: DANIILIDIS, K.; MARAGOS, P.; PARAGIOS, N. (Ed.). **Computer Vision – ECCV 2010**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 778–792. ISBN 978-3-642-15561-1.

CAMPOLO, C.; MOLINARO, A. On vehicle-to-roadside communications in 802.11p/wave vanets. In: **2011 IEEE Wireless Communications and Networking Conference**. [S.l.: s.n.], 2011. p. 1010–1015. ISSN 1558-2612.

E.RUBLEE et al. Orb: An efficient alternative to sift or surf. In: **2011 International Conference on Computer Vision**. [S.l.: s.n.], 2011. p. 2564–2571. ISSN 2380-7504.

FELICE, M. D. et al. A distributed beaconless routing protocol for real-time video dissemination in multimedia vanets. **Computer Communications**, v. 58, p. 40 – 52, 2015. ISSN 0140-3664. Special Issue on Networking and Communications for Smart Cities.

FERRONATO, J.; TRENTIN, M. Simulação de vanets utilizando ferramentas omnet++, sumo e veins. In: **II Simpósio de Informática do IFSul 2015**. [S.l.: s.n.], 2015.

FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. **Commun. ACM**, ACM, New York, NY, USA, v. 24, n. 6, p. 381–395, jun. 1981. ISSN 0001-0782. Disponível em: <http://doi-acm-org.ez26.periodicos.capes.gov.br/10.1145/358669.358692>.

GHOSH, D.; KAABOUCH, N. A survey on image mosaicing techniques. **Journal of Visual Communication and Image Representation**, v. 34, p. 1 – 11, 2016. ISSN 1047-3203. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1047320315002059>.

GROUP, I. P. W. **IEEE P2020 Automotive Imaging**. 2018. <https://site.ieee.org/sagroups-2020>.

GVOZDEN, G.; GRGIC, S.; GRGIC, M. Blind image sharpness assessment based on local contrast map statistics. **Journal of Visual Communication and Image Representation**, v. 50, p. 145 – 158, 2018. ISSN 1047-3203. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1047320317302262>.

HARSCH, C.; FESTAG, A.; PAPADIMITRATOS, P. Secure position-based routing for vanets. In: **2007 IEEE 66th Vehicular Technology Conference**. [S.l.: s.n.], 2007. p. 26–30. ISSN 1090-3038.

HARTENSTEIN, H.; LABERTEAUX, L. P. A tutorial survey on vehicular ad hoc networks. **IEEE Communications Magazine**, v. 46, n. 6, p. 164–171, June 2008. ISSN 0163-6804.

HOEBEKE, J. et al. An overview of mobile ad hoc networks: Applications and challenges. **JOURNAL-COMMUNICATIONS NETWORK**, v. 3, p. 60–66, 07 2004.

IEEE, I. standards fact sheets of. **Application Areas**. 2009. <https://www.standards.its.dot.gov/LearnAboutStandards/ApplicationAreas>.

IEEE, I. standards fact sheets of. **ITS standards fact sheets of IEEE**. 2009. <https://www.standards.its.dot.gov/factsheets/factsheet/80>.

ITU-T. **Subjective video quality assessment methods for multimedia application**. [S.l.], 2008.

JAKUBIAK, J.; KOUCHERYAVY, Y. State of the art and research challenges for vanets. In: **2008 5th IEEE Consumer Communications and Networking Conference**. [S.l.: s.n.], 2008. p. 912–916. ISSN 2331-9852.

JEVTIĆ, N.; MALNAR, M. Implementation of ETX metric within the AODV protocol in the NS-3 simulator. **Telfor Journal**, Centre for Evaluation in Education and Science (CEON/CEES), v. 10, n. 1, p. 20–25, 2018. Disponível em: <https://doi.org/10.5937/telfor1801020j>.

JIANG, D.; DELGROSSI, L. Ieee 802.11p: Towards an international standard for wireless access in vehicular environments. In: **VTC Spring 2008 - IEEE Vehicular Technology Conference**. [S.l.: s.n.], 2008. p. 2036–2040. ISSN 1550-2252.

JOHNSON, D. B.; MALTZ, D. A. Dynamic source routing in ad hoc wireless networks. In: ____. **Mobile Computing**. Boston, MA: Springer US, 1996. p. 153–181. Disponível em: <https://doi.org/10.1007/978-0-585-29603-6_5>.

KAUR, J.; JYOTI, D. Image quality assessment techniques pn spatial domain. In: . [S.l.: s.n.], 2011.

KIM, T.-H. et al. An effective data dissemination in vehicular ad-hoc network. In: VAZÃO, T.; FREIRE, M. M.; CHONG, I. (Ed.). **Information Networking. Towards Ubiquitous Networking and Services**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 295–304. ISBN 978-3-540-89524-4.

KUROSE, J. F.; ROSS, K. W. **Computer Networking: A Top-Down Approach**. 5th. ed. USA: Addison-Wesley Publishing Company, 2009. ISBN 0136079679, 9780136079675.

LEE, D.; YOON, J.; LIM, S. Image stitching using multiple homographies estimated by segmented regions for different parallaxes. In: **2017 International Conference on Vision, Image and Signal Processing (ICVISP)**. [S.l.: s.n.], 2017. p. 71–75.

LEUTENEGGER, S.; CHLI, M.; SIEGWART, R. Y. Brisk: Binary robust invariant scalable keypoints. In: **2011 International Conference on Computer Vision**. [S.l.: s.n.], 2011. p. 2548–2555. ISSN 2380-7504.

LI, F.; WANG, Y. Routing in vehicular ad hoc networks: A survey. **IEEE Vehicular Technology Magazine**, v. 2, n. 2, p. 12–22, June 2007. ISSN 1556-6072.

LINDEBERG, M. et al. Challenges and techniques for video streaming over mobile ad hoc networks. **Multimedia Systems**, v. 17, n. 1, p. 51–82, Feb 2011. ISSN 1432-1882.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. **International Journal of Computer Vision**, v. 60, n. 2, p. 91–110, Nov 2004. ISSN 1573-1405. Disponível em: <https://doi.org/10.1023/B:VISI.0000029664.99615.94>.

MIKOLAJCZYK, K. et al. A comparison of affine region detectors. **Int. J. Comput. Vision**, Kluwer Academic Publishers, USA, v. 65, n. 1–2, p. 43–72, nov. 2005. ISSN 0920-5691. Disponível em: <https://doi.org/10.1007/s11263-005-3848-x>.

MITTAL, A.; MOORTHY, A. K.; BOVIK, A. C. No-reference image quality assessment in the spatial domain. **IEEE Transactions on Image Processing**, v. 21, n. 12, p. 4695–4708, Dec 2012. ISSN 1057-7149.

MITTAL, A.; SOUNDARARAJAN, R.; BOVIK, A. C. Making a “completely blind” image quality analyzer. **IEEE Signal Processing Letters**, v. 20, n. 3, p. 209–212, March 2013. ISSN 1070-9908.

MOREELS, P.; PERONA, P. Evaluation of features detectors and descriptors based on 3d objects. **Int. J. Comput. Vision**, Kluwer Academic Publishers, USA, v. 73, n. 3, p. 263–284, jul. 2007. ISSN 0920-5691. Disponível em: <https://doi.org/10.1007/s11263-006-9967-1>.

NEWS, G. **Brasil perde R$ 267 bilhões por ano com congestionamentos**. 2018. <https://g1.globo.com/globonews/noticia/2018/08/07/brasil-perde-r-267-bi-por-ano-com-congestionamentos.ghtml>.

OLARIU, S.; WEIGLE, M. C. **Vehicular Networks: From Theory to Practice**. 1. ed. [S.l.]: Chapman & Hall/CRC, 2009. ISBN 1420085883, 9781420085884.

OMNET++. **Objective Modular Network Testbed in C++**. 2020.

PATEL, A.; KAUSHIK, P. Improving qos of vanet using adaptive cca range and transmission range both for intelligent transportation system. **Wireless Personal Communications**, v. 100, n. 3, p. 1063–1098, Jun 2018. ISSN 1572-834X.

P.NAND; SHARMA, S. C. Probability based improved broadcasting for aodv routing protocol. In: **2011 International Conference on Computational Intelligence and Communication Networks**. [S.l.: s.n.], 2011. p. 621–625.

RANJAN, P.; AHIRWAR, K. K. Comparative study of vanet and manet routing protocols. In: . [S.l.: s.n.], 2011.

ROSTEN, E.; DRUMMOND, T. Machine learning for high-speed corner detection. In: LEONARDIS, A.; BISCHOF, H.; PINZ, A. (Ed.). **Computer Vision – ECCV 2006**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 430–443. ISBN 978-3-540-33833-8.

SANTOSH, K. C. Let vehicles talk with images for smart vehicular communication system. In: SANTOSH, K. C.; HEGADI, R. S. (Ed.). **Recent Trends in Image Processing and Pattern Recognition**. Singapore: Springer Singapore, 2019. p. 17–26. ISBN 978-981-13-9181-1.

SILVA, L. H. C.; MINI, R. A. F.; CUNHA, F. D. Um protocolo de disseminação de dados geo-orientado em redes veiculares. In: **Anais do I Workshop de Computação Urbana**. Porto Alegre, RS, Brasil: SBC, 2017. ISSN 2595-2706. Disponível em: <https://sol.sbc.org.br/index.php/courb/article/view/2587>.

SMIDA, E. B.; FANTAR, S. G.; YOUSSEF, H. Video streaming forwarding in a smart city's vanet. In: **2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA)**. [S.l.: s.n.], 2018. p. 1–8. ISSN 2163-2871.

SOMMER, C.; GERMAN, R.; DRESSLER, F. Bidirectionally coupled network and road traffic simulation for improved ivc analysis. **IEEE Transactions on Mobile Computing**, v. 10, n. 1, p. 3–15, Jan 2011. ISSN 1536-1233.

SUMO. **Simulation of Urban MObility**. 2020.

SZELISKI, R. **Computer Vision: Algorithms and Applications**. 1st. ed. Berlin, Heidelberg: Springer-Verlag, 2010. ISBN 1848829345, 9781848829343.

TAREEN, S. A. K.; SALEEM, Z. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In: **2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)**. [S.l.: s.n.], 2018. p. 1–10.

TIMES, T. N. Y. **Stuck and Stressed: The Health Costs of Traffic**. 2019. <https://www.nytimes.com/2019/01/21/upshot/stuck-and-stressed-the-health-costs-of-traffic.html>.

TOOR, Y. et al. Vehicle ad hoc networks: applications and related technical issues. **IEEE Communications Surveys Tutorials**, v. 10, n. 3, p. 74–88, Third 2008. ISSN 1553-877X.

TSENG, Y.-C.; NI, S.-Y.; SHIH, E.-Y. Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network. **IEEE Transactions on Computers**, v. 52, n. 5, p. 545–557, May 2003. ISSN 1557-9956.

UZCATEGUI, R. A.; SUCRE, A. J. D.; ACOSTA-MARUM, G. Wave: A tutorial. **IEEE Communications Magazine**, v. 47, n. 5, p. 126–133, May 2009. ISSN 0163-6804.

VISION, O. S. C. **BFMatcher Class Reference**. 2018. <https://docs.opencv.org/3.4.5/d3/da1/classcv_1_1BFMatcher.html#details>.