

Transmissão de Vídeo em Tempo Real em Redes AD HOC

Anderson Andrei¹, Patrick Abrahão¹, Roger Immich², Alfredo Goldman¹

¹Instituto de Matemática e Estatística
Universidade de São Paulo(USP)

²Instituto de Computação
Universidade Estadual de Campinas (UNICAMP)

{anderson.andrei.silva, patrick.menani}@usp.br,
roger@lrc.ic.unicamp.br, gold@ime.usp.br

Abstract. *In the next few years is expected a considerable growth of video transmission in wireless networks, this is especially true considering mobile devices. In the light of this, the main goal of this work is to perform a quantitative and qualitative analysis of the real-time video transmission between mobile devices. This is important to better understand all the details involved in this process. In order to do that, several methods of package transmission of live video stream were analyzed. Using a network simulator OMNeT++ and the INET Framework, these methods were assessed and classified according to their qualities and scalability. A comparison between the protocols TCP, UDP, and the transmission technique DASH is also presented. So, we did analysis with the metrics of delay in the delivery of the packages, among of KiB received by the receiver host and the package loss and required rate.*

Resumo. *Nos próximos anos é esperado um grande crescimento da transmissão de vídeos em redes sem fio, especialmente em dispositivos móveis. Baseando-se nesta crescente demanda, o principal objetivo deste trabalho é o de realizar uma análise quantitativa e qualitativa sobre a transmissão de vídeo em tempo real entre dispositivos móveis. Com isto, é esperado obter-se um melhor entendimento de alguns aspectos ligados à transmissão de dados. Para tal, foram analisadas algumas formas de transmissão de pacotes na rede para vídeo ao vivo. Através de simulações foi possível realizar comparações entre as diversas formas afim de classificá-las quanto sua qualidade e a escalabilidade na rede. As comparações foram feitas entre os protocolos TCP, UDP e a técnica de transmissão DASH através de simulações feitas no OMNeT++ e o INET Framework. Assim, fizemos análises com as métricas de atraso na entrega dos pacotes, quantidade total de KiB recebidos pelo dispositivo destino e a taxa de pacote recebidos e requisitados.*

1. Introdução

Nos últimos anos houve um grande aumento na utilização de aplicações de vídeo. Estas aplicações estão presentes nas redes sociais, serviços de vídeo sob demanda e até em jogos interativos. Para aumentar ainda mais a demanda, cada vez mais são produzidos mais e melhores dispositivos móveis [1], colocando um grande poder de processamento nas mãos dos usuários. Isto permite o fácil acesso à produção, consumo e disponibilização

de conteúdos multimídia. Todas as facilidades levam a um considerável aumento no tráfego de rede. Somente para citar um exemplo, de acordo com a Cisco, até 2021 o tráfego global de toda a Internet deverá ser 127 vezes maior do que foi em 2005. De todo este tráfego, é esperado que a transmissão de vídeo represente mais de 82% do total [2]. Levando-se em consideração este grande crescimento na demanda pela transmissão de vídeo, este trabalho visa estudar a transmissão de dados em tempo real em redes veiculares tendo como maior motivação o cenário de veículos autônomos. Redes veiculares tem apresentado atualmente vantagens como controle eficiente de tráfego, maior segurança e possibilidade de desenvolvimento de novos recursos [3]. Esse ambiente conhecido como redes veiculares *AD-HOC* (*VANETs*) é composto por uma estrutura capaz de fornecer dados para veículos que trafegam em uma determinada área, e outra estrutura que configura o conjunto de veículos que utilizarão o serviço.

Dada a complexidade de trabalhar com esse ambiente, em um primeiro momento, não trabalharemos com redes veiculares. Portanto, reduzimos nosso cenário de estudo para redes móveis *AD-HOC*. Estas redes são compostas por dispositivos móveis que não necessitam de uma infra-estrutura de rede por trás e possuem movimentação, tendo assim cenários dinâmicos. Dessa forma, vamos estudar pontos de qualidade de serviço e de experiência do usuário para a transmissão de vídeo em tempo real em redes *AD-HOC*. Vamos fazer comparações entre dois protocolos de transmissão, *TCP* e *UDP*, e o *DASH*, uma técnica aplicada no primeiro deles que permite a mudança da qualidade da transmissão de vídeo de acordo com a qualidade da rede. Utilizaremos o simulador de redes *OMNeT++* [4] com a extensão *INET Framework* [5] que nos possibilita trabalhar também com cenários móveis e redes sem fio. A partir dessas simulações vamos analisar requisição e perda de pacote, o atraso na entrega deles e a quantidade total de *KiB* recebidos pelo dispositivo destino para analisar qual deles pode ser melhor em determinados cenários. Nesse trabalho são apresentados alguns conceitos sobre redes *AD-HOC*, a técnica de transmissão *DASH* e os critérios de avaliação que serão utilizados na análise. Também serão apresentados alguns trabalhos relacionados, os dados obtidos e suas respectivas análises. Por fim, será apresentada a conclusão e trabalhos futuros.

2. Conceitos fundamentais

A transmissão de vídeos em tempo real é algo complexo que demanda o estudo da estrutura da rede que será utilizada, das ferramentas responsáveis pela transmissão do vídeo, sua qualidade e integridade, entre diversos outros conceitos [6]. Assim sendo, serão apresentados alguns conceitos fundamentais utilizados no desenvolvimento deste trabalho.

2.1. Redes AD HOC

Uma rede móvel *AD-HOC*, do inglês *MANET* e também conhecida como *WANET* (*Wireless Ad Hoc Network*) é uma rede composta por um conjunto de dispositivos de rede sem fio que não precisam, necessariamente, de uma infra estrutura de rede pré existente. É uma rede descentralizada e auto organizada onde as comunicações são feitas de forma multi-saltos, ou seja, partem de um dispositivo fonte para um dispositivo destino sendo transmitidas através de dispositivos que estão localizados entre esses dois. De acordo com a mobilidade de tais dispositivos, estas redes apresentam características altamente dinâmicas, o que dificulta a análise de qualidade de serviço e experiência do usuário final [7].

2.2. Técnica de transmissão DASH

Essa é uma técnica aplicada no protocolo TCP e permite ao cliente redimensionar e optar pelo melhor *bitrate*. Dependendo das limitações de recursos oferece uma maneira eficiente de proporcionar a melhor qualidade possível para o usuário. Ou seja, essa técnica faz com que a qualidade do vídeo diminua caso a qualidade da rede seja baixa ou fique assim por algum motivo. Da mesma forma, se a qualidade da rede é alta o suficiente, a qualidade do vídeo transmitido será proporcionalmente alto. Logo é mais garantida a transmissão do vídeo em tempo real em maiores circunstâncias [8] .

2.3. Critérios de comparação

Para analisarmos os dados obtidos utilizaremos como critério de comparação requisição e perda de pacotes, quantidade de *KiB* recebidos pelo dispositivo destino e o atraso na entrega deles. O primeiro é o número de pacotes requisitados e taxa de perda deles dentro da trajetória até seu destino final, associamos essa métrica objetiva à perda de informação, no nosso caso, trechos de vídeo. Assim, interpretamos que quanto mais pacotes forem perdidos pior é a reprodutibilidade do vídeo transmitido. O segundo trata da qualidade do vídeo recebido pelo cliente, quanto mais informação for recebida através de cada pacote maior é a qualidade do vídeo reproduzido pelo dispositivo final. O último é o quanto de atraso ocorreu na entrega dos pacotes, refletindo na reprodução em tempo real. Assim, interpretamos que todas essas métricas podem ser indicativos da QoE do usuário final.

3. Trabalhos relacionados

Esse trabalho está inserido no projeto INCT da Internet do Futuro para Cidades Inteligentes : "*Improving fog computing techniques to allow a better quality of experience in vehicular networks*", e no mesmo contexto da pesquisa apresentada foram encontramos diversos trabalhos relacionados. Isto demonstra que o trabalho desenvolvido está inserido em um assunto importante e com um grande potencial de crescimento.

O aumento do uso de *cloud computing* requer atenção por ainda envolver vários pontos não resolvidos como latência não confiável, falta de mobilidade e pontos de localização para disponibilização da rede. *Fog computing* ou *edge computing* aparece então nesse cenário provendo recursos e serviços maleáveis para os usuários finais através da borda da rede, enquanto, em conjunto, a *cloud computing* passa a ser responsável pela rede central, solucionando assim o problema de falta de mobilidade. Tratando a questão da latência, a *fog computing* fornece dinamismo e maior confiabilidade de conexão. Por outro lado, para melhor aproveitar recursos como capacidade e banda, é preciso analisar como e onde pré armazenar dados para tornar mais prática a computação dos dados e esse tem sido um dos desafios [3]. Nesse mesmo segmento são estudadas decisões quanto a troca de informação utilizando máquinas virtuais, análises do gerenciamento de recursos e da comunicação entre servidor e veículos ao longo de estradas [9]. Essa comunicação por parte do servidor e do veículo pode falhar em algum momento ou até não alcançar um determinado veículo e então é desejado que o mesmo possa se comunicar com outros que tenham esta conexão estabelecida, mantendo assim o fluxo de dados ao longo de um percurso [10].

Além disso, também são estudadas configurações e composições de vídeos. Esses são formados por *frames* e utilizados de formas diferentes dependendo do tipo de *codec*

utilizado. Esses *frames* possuem diferenças em suas composições podendo ser classificados em três tipos diferentes de prioridades e ainda se relacionam dependentemente entre si. Logo, ocasionalmente a perda de um *frame* em específico pode afetar um longo trecho de vídeo, o que é diretamente perceptível ao usuário, e então é um sério problema de qualidade de experiência. Por isso são estudados e desenvolvidos métodos de como tratar e evitar tais problemas, também aplicados em transmissão de vídeo em tempo real em redes *AD-HOC*. Assim sendo, conceitos como esses, técnicas para seu funcionamento e as especificidades da rede que permite esse uso tem sido foco de estudos [7].

4. Cenários experimentais

Nessa primeira abordagem será simulada a transmissão de vídeo em tempo real em uma rede *AD-HOC*. Esta rede será composta por vários dispositivos móveis e todos eles possuem as mesmas características, porém variam entre três configurações distintas, são elas: emissor, receptor e intermediador. O primeiro é aquele que envia os dados, o segundo é quem deve recebê-los e o último é aquele que repassa os dados construindo um caminho ligando o emissor até o receptor. Existem diferentes protocolos que atuam de maneiras distintas para o funcionamento de uma rede. Para o envio de pacotes são necessários protocolos que encapsulam as informações antes do envio e existem outros protocolos que são responsáveis pelo roteamento dos pacotes. Neste trabalho, serão utilizados para a transmissão dos dados os protocolos *TCP* e *UDP* e ainda a técnica *DASH*, aplicada em cima do *TCP*. E para o roteamento utilizaremos o protocolo *AODV*. A utilização do *DASH* nesse trabalho é uma adaptação da implementação disponível no *GitHub* de Navarro Joaquin [11].

4.1. Modelo estudado

Os cenários implementados são compostos por um dispositivo emissor, *hostA*. Um dispositivo receptor, *hostB*, e n dispositivo intermediadores, *hostR*(1, 2, 3, ... n). Os *hosts A* e *B* terão suas posições iniciais fixadas dentro do cenário sendo respectivamente (782.86, 544.91) e (145.15, 242.71) em todas as simulações, já os *hosts R* serão posicionados aleatoriamente. A quantidade de *hosts R* também será alterada entre as quantidades 50, 100 e 150. Todos os dispositivos se movimentam em duas velocidades, 3 e 6 m/s, gerando diferentes caminhos para o envio dos dados e consequentemente, dinamismo na rede. A movimentação ocorre de forma livre com a única restrição sendo a área total do experimento, seguindo o padrão *MassMobility* implementado no simulador [5]. A área do cenário de testes é de 1000 x 700 m e no total, serão avaliadas 6 variações de cenários distintas, que são, três quantidades diferentes de dispositivos com movimentação lenta e as mesmas três quantidades diferentes de dispositivos com movimentação rápida. A duração do vídeo transmitido nas simulações é de 90s. O tamanho dos pacotes do *UDP* e *TCP* foram fixados em 60 *KiB*, para o *DASH* colocamos o mais próximo disso como tamanho inicial, já que o mesmo se modifica no decorrer da simulação.

5. Análise de performance e resultados

Foram executadas 10 simulações para cada modelo de cenário descrito acima nos protocolos *TCP*, *UDP* e *DASH*, sendo então, 60 simulações para cada um deles, totalizando 180 amostras de resultados. A partir desses dados foi obtida a média e alguns desvios padrões para as métricas de atraso, total de *KiB* recebidos pelo *hostB* e quantidade

de pacotes requisitados e sua taxa de perda. Também foram gerados histogramas para o atraso e estão disponíveis no repositório do projeto [12], seus nomes são compostos por *PROTOCOLO_(velocidade)(número de dispositivos)* e seus dados foram traduzidos em algumas das tabelas que serão exibidas abaixo.

5.1. Delay

A partir da tabela 1 é possível observar que o *UDP* apresenta valores próximos de zero, variando, nos cenários de velocidade 3 m/s entre 0.41 e 0.6, e nos cenários de 6 m/s entre nan (valor minimamente pequeno) e 2.23. A maior surpresa foi o maior valor ocorrer para a velocidade de 6 m/s com 150 hosts, mas no geral, o *UDP* se mostrou muito bem em relação a essa métrica. Já o *TCP*, possui em todos os casos valores maiores que o *UDP*. Para os cenário de velocidade de 3 m/s varia entre 2.20 e 3.16, melhorando seu rendimento conforme o aumento no número de dispositivos, enquanto, para os cenários com velocidade 6 m/s varia entre 2.77 e 5.07, tendo também seu maior valor no cenário de maior velocidade e maior quantidade de dispositivos. Esses valores podem ser maiores que os do *UDP* pelo fato dele não fazer verificações, como as de destinatário dos pacotes por exemplo, antes de enviá-los, enquanto o *TCP* faz. O *DASH* aumenta um pouco seu intervalo de valores em relação ao *TCP*, variado entre 3.66 e 5.53 na velocidade de 3 m/s, e 4.98 e 5.84 na velocidade de 6 m/s. O que pode ser percebido é que a variação entre esses valores é menor no caso com maior velocidade, e pode ser assumido que os valores do *DASH* são um pouco maiores que os do *TCP* devido a maior quantidade de processos de controle da rede que o *DASH* faz.

Tabela 1. UDP, TCP, DASH delay

Parameters	Velocity of hosts : 3m/s			Velocity of hosts : 6m/s		
	50 hosts	100 hosts	150 hosts	50 hosts	100 hosts	150 hosts
<i>UDP</i>						
Mean	0.49	0.60	0.41	0.94	nan	2.23
Stddev	0.55	0.81	0.40	1.14	nan	1.67
<i>TCP</i>						
Mean	3.16	2.91	2.20	3.83	2.77	5.07
Stddev	4.96	3.59	1.90	5.91	2.81	5.13
<i>DASH</i>						
Mean	5.53	3.82	3.66	5.84	4.98	5.55
Stddev	7.56	3.65	3.12	7.44	5.84	4.97

5.2. Requisição e Perda de Pacotes

A tabela 2, abaixo, apresenta o número de pacotes requisitados e a taxa de perda dos mesmos, com variação na quantidade de dispositivos. A partir dela foi observado que o *UDP* requisita, em todos os casos, valores próximos de 90 pacotes, e como simulamos a transmissão de 90s de vídeo, apontamos que o *UDP* consiga transmitir 1s de vídeo por pacote nos cenários simulados. Os resultados mostram uma perda de cerca de 50% dos pacotes em quase todos os casos, exceto no cenário de 3 m/s com 150 dispositivos, 33.90%, mas ainda assim é um valor considerado alto pelo grupo. O *TCP* requisita em

quase todos os casos mais de 100 pacotes. Isto pode ser explicado devido a diferente forma dele gerenciar as entregas em relação ao *UDP*. Sendo assim, talvez não entregue 1s de vídeo por pacote, precisando requisitar mais deles. Quanto a taxa de perda de pacotes seu rendimento é bem melhor do que o *UDP* com valores variando entre 2.41 e 3.25% para os cenários de 3 m/s, mostrando uma melhora com o aumento do número de dispositivos, e para os cenários de 6 m/s se mantém entre 2.66 e 4.54%, mas sem apresentar melhora com o aumento da quantidade de dispositivos. O *DASH* apresenta o comportamento parecido com o *TCP*, fazendo até mais requisições de pacotes, variando entre 103 e 145. Seguindo o mesmo argumento do *TCP* em relação ao *UDP*, é possível justificar que o *DASH* faça mais requisições do que o *TCP* justamente por tentar garantir mais procedimentos, incluindo a alteração no tamanho dos pacotes. Em relação a perda de pacotes, este apresenta uma diminuição conforme o aumento do número de dispositivos nos cenários de 3m/s, variando entre 4.27 e 6.77%, e para 6m/s apresenta valores entre 6.05 e 7.01%. Nota-se que esses valores são maiores que os do *TCP*, mas ainda assim, bem menores do que os do *UDP*.

Tabela 2. UDP, TCP, DASH requisição e perda de pacotes

Parameters	Velocity of hosts : 3m/s			Velocity of hosts : 6m/s		
	50 hosts	100 hosts	150 hosts	50 hosts	100 hosts	150 hosts
<i>UDP</i>						
App Request	89.8	90.00	90.00	89.90	80.90	89.90
Pck Loss Rate (%)	46.00	54.22	33.90	67.40	51.91	60.62
<i>TCP</i>						
App Request	123.20	135.90	166.20	126.20	150.30	88.10
Pck Loss Rate (%)	3.25	2.94	2.41	3.17	2.66	4.54
<i>DASH</i>						
App Request	103.30	145.20	145.9	103.40	122.20	108.40
Pck Loss Rate (%)	6.77	4.27	4.73	6.38	6.05	7.01

5.3. Total de *KiB* recebidos

Para essa análise foi levado em consideração a estrutura dos pacotes em cada um dos protocolos / técnica, todos eles são compostos por cabeçalhos e os dados propriamente ditos. Os mesmos possuem tamanhos de cabeçalhos diferentes, o que levaria ao fato de cada um deles poder entregar menos *KiB* de dados, de vídeo no nosso caso, por pacotes. Mas no simulador utilizado o tamanho do cabeçalho é desconsiderado para o *TCP* e *DASH*, tendo influência apenas no *UDP*. Para realizar uma comparação mais justa entre os protocolos, foram desconsiderados os tamanhos dos cabeçalhos, analisando somente o *payload*. A tabela 3, abaixo, apresenta a média desse total acompanhado por seu desvio padrão, com variação na quantidade de dispositivos. A partir dela foi observado que o *UDP*, na menor velocidade, chega a enviar mais *KiB* do que em qualquer caso na velocidade mais rápida. Para 3 m/s ele transmite mais de 1953.1 *KiB* em todos os casos, variando entre 2414.0 e 3486.3 *KiB*. Já com a velocidade de 6 m/s, o maior valor foi de 2220.2 *KiB*, variando entre 1631.6 e 2220.2 *KiB*. Assim sendo, é possível apontar que o aumento de velocidade pode interferir no total de *KiB* transmitidos utilizando esse protocolo. No caso do *TCP*, esses valores aumentam em uma ordem de grandeza em relação

ao *UDP*, ou seja, são em torno de 10x maiores. Foi verificado que para o cenário com velocidade de 3 m/s os valores aumentam de acordo com o aumento do número de dispositivos, variando entre 14281.3 e 19320.3 *KiB*. Para o cenário de 6 m/s não se mantém esse comportamento, e os valores em média são menores do que os com a velocidade de 3 m/s, variando entre 14436.9 e 17457.0 *KiB*. É possível notar que para esse caso a velocidade também não favorece essa métrica. No caso do *DASH*, a ordem de grandeza desses valores é igual a do *UDP*, mas ainda todos são maiores. Para o caso de 3 m/s, variam entre 6071.7 e 9056.5 *KiB*, não proporcionalmente ao aumento do número de dispositivos. Com o mesmo comportamento, podemos analisar os cenários com velocidade de 6 m/s, variando entre 6948.8 e 7412.9 *KiB*. Isso demonstra que a qualidade da rede não é tão boa e o *DASH* pode estar diminuído a quantidade de *KiB* transmitidos, diminuindo também a qualidade do vídeo.

Tabela 3. UDP, TCP, DASH total de *KiB* recebidos

Parameters	Velocity of hosts : 3m/s			Velocity of hosts : 6m/s		
	50 hosts	100 hosts	150 hosts	50 hosts	100 hosts	150 hosts
<i>UDP</i>						
Mean (<i>KiB</i>)	2841.8	2414.0	3486.3	1631.6	2220.2	1997.6
Stddev	1084.5	1319.0	833.9	962.7	1275.4	1040.4
<i>TCP</i>						
Mean (<i>KiB</i>)	14281.6	15769.5	19320.3	14632.8	17457.0	14436.9
Stddev	7463.3	8435.2	9794.2	9292.4	9983.0	0
<i>DASH</i>						
Mean (<i>KiB</i>)	6071.7	9056.5	8854.3	6948.8	7412.9	7064.4
Stddev	2387.6	3254.0	1743.9	3427.7	3193.6	1984.4

6. Conclusão

O serviço com *UDP* tem um maior desempenho em relação ao atraso da rede, porém é necessário analisar se essa diferença em relação ao *TCP/DASH* afetaria realmente a transmissão, pois ele pode não ser nem perceptível ao usuário, considerando que é uma transmissão de vídeo, assumimos que a continuidade do fluxo de dados é mais importante. Em relação ao número de pacotes enviados e recebidos o *UDP* possui alta perda de pacotes durante a transmissão, fazendo com que a análise *atraso x perda de pacotes* em relação ao *TCP/DASH* não fique tão discrepante, tendo então que mesmo com pouco atraso na entrega, não é possível manter um fluxo de dados constante pro caso do *UDP*.

O *TCP* envia mais dados durante a transmissão, onde dados são *KiB*, mas talvez parte desses dados sejam repetidos devido ao processos para evitar falhas no envio/recebimento dos pacotes. O *DASH* envia uma quantidade com uma ordem de grandeza a menos que o *TCP* mas ainda assim, mais do que o *UDP*, isto pode indicar que então, mesmo com um certo atraso, ele ainda envia mais dados durante a transmissão garantindo ainda pouca perda de pacotes. O *TCP* age dessa forma, mas ainda não está claro a que custos ele envia mais dados.

Portanto, é possível dizer que as vantagens de características em aplicações que o *DASH* não comprometem muito quanto ao recebimento dos pacotes enviados pelo serviço

de servidor, ou seja, eles entregam um número maior ao que o *UDP* entregaria na rede, mas nos mostraram fazer o recebimento e consequentemente o fluxo dos dados ser mais garantido, o que nos cenários avaliados, na transmissão de vídeo em tempo real é consideravelmente mais importante.

Dada a magnitude do projeto, a continuidade deste trabalho poderá contemplar a implementação de : Análise multi-saltos do envio dos pacotes na rede; Análise da entrega de *frames* de vídeo; Integração com o *SUMO* para melhorar a mobilidade urbana; Servidores especializados para redes veiculares, ou seja, a inclusão de *FOG Computing*; Trabalhar com redes veiculares propriamente ditas; Integração com o *Veins* para simular redes veiculares.

Agradecimentos

Os autores gostariam de agradecer o apoio financeiro concedido através do projeto INCT Internet do Futuro, à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) através do processo 2018/02204-6, à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Referências

- [1] Adobe Digital Index. A look at the 2017 adobe mobile maturity survey. Technical report, Adobe, 2017.
- [2] Cisco. White paper: Cisco VNI forecast and methodology, 2016-2021. Technical report, Cisco, September 2017.
- [3] Shanhe Yi, Cheng Li, and Qun Li. A survey of fog computing: Concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*, Mobidata '15, pages 37–42. ACM, New York, NY, USA, 2015.
- [4] Omnet++, <https://www.omnetpp.org/>.
- [5] Inet framework, <https://inet.omnetpp.org/>.
- [6] Roger Immich, Eduardo Cerqueira, and Marilia Curado. *Mechanisms for Resilient Video Transmission*. PhD thesis, University of Coimbra. <http://www.lrc.ic.unicamp.br/roger/phdThesis.pdf>, October 2017.
- [7] Roger Immich, Eduardo Cerqueira, and Marilia Curado. Efficient high-resolution video delivery over vanets. *Wireless Networks*, Feb 2018.
- [8] Mpeg-dash vs. apple hls vs. microsoft smooth streaming vs. adobe hds.
- [9] Hong Yao, Changmin Bai, Deze Zeng, Qingzhong Liang, and Yuanyuan Fan. Migrate or not? exploring virtual machine migration in roadside cloudlet-based vehicular cloud. *Concurr. Comput. : Pract. Exper.*, 27(18):5780–5792, December 2015.
- [10] Rong Yu, Yan Zhang, Stein Gjessing, W. Xia, and K. Yang. Toward cloud-based vehicular networks with efficient resource management. *IEEE Network Magazine*, 27(5):48–55, 2013.
- [11] Navarro joaquin - github, <https://github.com/navarrojoaquin/adaptive-video-tcp-omnet>.
- [12] Vehicularnetworksic - github, <https://github.com/andersonandrei/vehicularnetworksic/>.