Universidade Federal de Juiz de Fora Programa de Pós-Graduação em Engenharia Elétrica

Vinicius Ferreira Vidal

Reconstrução 3D de Subestações de Energia para Monitoramento Remoto com Arquitetura *Edge-Fog* com estudo de integração ao Contexto 5G

Tese de Doutorado

Juiz de Fora

2023

VINICIUS FERREIRA VIDAL

Reconstrução 3D de Subestações de Energia para Monitoramento Remoto com Arquitetura *Edge-Fog* com estudo de integração ao Contexto 5G

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora, na área de concentração em Sistemas de Energia, como requisito para obtenção do título de Doutor em Engenharia Elétrica.

Orientador: Prof. Dr. Leonardo de Mello Honório
Co-orientador: Prof. Dr. Mario Antônio Ribeiro Dantas

Juiz de Fora
2023

Vinicius Ferreira Vidal

Reconstrução 3D de Subestações de Energia para Monitoramento Remoto com Arquitetura Edge-Fog com estudo de integração ao Contexto 5G

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: Sistemas de Energia Elétrica

Aprovada em 31 de maio de 2023.

BANCA EXAMINADORA

Prof. Dr. Leonardo de Mello Honório - Orientador

Universidade Federal de Juiz de Fora

Prof. Dr. Mario Antonio Ribeiro Dantas - Coorientador

Universidade Federal de Juiz de Fora

Prof. Dr. Alexandre Bessa dos Santos

Universidade Federal de Juiz de Fora

Prof. Dr. Delfim Soares Junior

Universidade Federal de Juiz de Fora

Prof. Dr. Diego Barreto Haddad

Centro Federal de Educação Tecnológica Celso Suckow da Fonseca

Profa. Dra. Marley Maria Bernardes Rebuzzi Vellasco

Pontifícia Universidade Católica do Rio de Janeiro

Juiz de Fora, 29/05/2023.

Documento assinado eletronicamente por **Leonardo de Mello Honorio**, **Professor(a)**, em 01/06/2023, às 10:17, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Mario Antonio Ribeiro Dantas**, **Professor(a)**, em 01/06/2023, às 11:33, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Diego Barreto Haddad**, **Usuário Externo**, em 01/06/2023, às 13:23, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Delfim Soares Junior**, **Professor(a)**, em 01/06/2023, às 15:45, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Alexandre Bessa dos Santos**, **Professor(a)**, em 01/06/2023, às 18:04, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Marley Maria Bernardes Rebuzzi Vellasco**, **Usuário Externo**, em 06/06/2023, às 02:29, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no Portal do SEI-Ufjf (www2.ufjf.br/SEI) através do ícone Conferência de Documentos, informando o código verificador **1303501** e o código CRC **7F549E89**.

Dedico este trabalho à minha família, em especial meu avô Francisco, minha esposa, meus amigos e professores que foram essenciais e de muitas maneiras ajudaram para que fosse possível a concretização do mesmo.

AGRADECIMENTOS

Ao meu pai Luiz Fernando e minha mãe Maria da Consolação, pela educação, dedicação e base para poder desenvolver meus estudos da melhor forma possível.

À minha irmã Larissa por ajudar em todos os meus trabalhos.

À minha esposa Fernanda pelo companheirismo e apoio nessa trajetória.

Ao meu avô Francisco pela sabedoria de anos.

A todos os meus familiares pelo incentivo e dedicação durante o período da pós-graduação.

Ao meu orientador, professor Leonardo Honório, e ao meu coorientador, Professor Mario Dantas, pela atenção e tempo dedicados ao trabalho.

A todos os amigos do GRIn, pelos bons momentos e ajuda durante todo o processo.

Aos bons amigos, pelo apoio e incentivo no curso e pelos bons momentos durante esses anos.

"Pensa problema não."

(Avô Francisco.)

RESUMO

A necessidade de automação e monitoramento remoto em subestações de energia vem sendo um tópico crescente na literatura e aplicações práticas, muitas vezes se tratando de uma tarefa tediosa em um ambiente energizado, de risco à saúde do operador, ou até mesmo de difícil acesso. Uma forma de abordagem para solução deste problema consiste na reprodução do cenário e seus componentes em ambiente virtual, por meio de reconstrução 3D dos entes reais da subestação. Atualmente, esse escaneamento é realizado em grande parte de forma não otimizada, contando com processamento embarcado ou somente aquisição de dados para serem trabalhados de maneira *offline* na nuvem. Essas aplicações reduzem, ou até mesmo inviabilizam a requisição em tempo real do estado dos componentes da subestação, além de não serem escaláveis e eficientes. Para solucionar esse desafio, este trabalho propõe duas frentes de solução: o uso de técnicas clássicas otimizadas pelo processamento paralelo de dados de nuvem de pontos por meio de um Algoritmo de Octree Iterativa; e uma arquitetura de computação *edge-fog*, a qual permite distribuição de carga computacional e armazenamento entre todos os entes da solução, estes por sua vez alocados em camadas, algo inédito na literatura para esta aplicação específica. O uso de rede de comunicação 5G é apresentado com uma breve discussão em como essa tecnologia pode beneficiar o processamento distribuído como um todo. Um robô foi desenvolvido para coletar os dados de imagens e nuvens de pontos. Todos os processos algorítmicos e matemáticos são alocados e detalhados para ambas as camadas *edge* e *fog*. Como resultado, o trabalho traz uma análise completa dos benefícios do uso da arquitetura frente a uma abordagem local e tradicional, baseada em *edge*. A escalabilidade da solução é demonstrada ao final, em ambiente real de aplicação, e nós concluímos o quanto uma rede de comunicação 5G iria beneficiar a solução neste cenário requerido para monitoramento remoto.

Palavras-chave: Escaneamento 3D, Monitoramento remoto, Arquitetura *edge-fog*, *Fog robotics*.

ABSTRACT

The need for automation and remote monitoring in power substations has been a growing topic in the literature and practical applications, often dealing with a tedious task in an energized environment, at risk to the operator's health, or even difficult to access. One way to approach this problem is to reproduce the scenario and its components in a virtual environment, through 3D reconstruction of the real entities of the substation. Currently, this scanning is performed largely in a non-optimized way, relying on embedded processing or only data acquisition to be worked offline in the cloud. These applications reduce, or even make it impossible to request, in real-time, the state of the substation components, in addition to not being scalable and efficient. Faced with this challenge, this work proposes two solution fronts: the use of classical techniques optimized by the parallel processing of point cloud data through an iterative octree; and an edge-fog computing architecture, which allows distribution of computational load and storage among all the entities in the solution, which in turn are allocated in processing layers (edge and fog), something unprecedented in the literature for this specific application. The 5G network application is presented along with a brief discussion on how this technology can benefit and allow the overall distributed processing. A robot is developed to collect data from images and point clouds. All algorithmic and mathematical processes are allocated and detailed for both layers. As a result, the work brings a complete analysis of the benefits of using architecture against a local and traditional approach, based on edge. The scalability of the solution is demonstrated at the end, in a real application environment, and we conclude with how much a private 5G network would benefit this solution in the required remote monitoring scenario.

Keywords: 3D scanning, Remote monitoring, Edge-fog architecture, Fog Robotics.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LiDAR *Light Detection and Ranging*

IoT *Internet of Things* (Internet das coisas)

ROS *Robot Operating System*

SLAM *Simultaneous Localization and Mapping* (Localização e Mapeamento Simultâneos)

SOR *Statistical Outlier Removal*

IMU *Inertial Measurement Unit* (Unidade de Medição Inercial)

FR *Fog Robotics* (Robótica em Névoa)

CR *Cloud Robotics* (Robótica em Nuvem)

SfM *Structure from Motion*

PCA *Principal Component Analysis*

ME Memória de Entrada

MS Memória de Saída

PU Por Unidade

kNN *K-Nearest Neighbors*

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, com o avanço da tecnologia e seguindo a tendência da indústria, a automação de estações de energia tem sido um tópico procurado tanto em termos acadêmicos quanto comerciais [1, 2]. Da mesma forma, o monitoramento remoto das mesmas surge como uma possibilidade cada vez mais viável, por se tratar de uma operação muitas vezes tediosa, repetitiva e arriscada para o operador, bem como de difícil acesso às instalações [3, 4].

Pode ser visto na literatura aplicações de monitoramento envolvendo estados de sensores e atuadores por meio de dispositivos *Internet of Things* (Internet das coisas) (IoT) [5, 6]. Em alguns casos, porém, o monitoramento das condições reais de equipamentos é relevante para sua operação segura [7]. Nesse caso, dados são aquisitados para performar a reconstrução 3D de ambientes e objetos, em sua grande maioria utilizando sensores LiDAR e câmeras [8, 9].

É uma tarefa desafiadora performar a reconstrução 3D de um ambiente relativamente extenso, que requer sensores precisos e vários pontos de vista para uma boa representação em ambiente virtual. Muitas aplicações atualmente dependem da coleta de dados local para pós-processamento, o qual pode ser custoso e por vezes não automatizado, impedindo a aplicação em tempo real [10]. Aplicações com processamento em nuvem, ou *cloud*, são vistas na literatura [11], porém as mesmas demandam grande capacidade de banda de rede, com latência considerável e menor confiabilidade devido à instabilidade de rede [12]. Outras aplicações têm seu processamento local, muitas vezes realizado nas bordas, as quais possuem restrições de capacidade computacional e energética [13]. Aplicações com coleta de dados também são vistas, não sendo possíveis em monitoramento de tempo real [8, 14].

Tratando-se mais especificamente de um ambiente de subestação de energia, por experiência prática dos autores ao longo de anos de projetos no ramo, o monitoramento remoto com reprodução de objetos reais e telepresença é de suma importância. Equipamentos como a chave seccionadora devem ser inspecionados de forma meticulosa para

identificar falhas no fechamento. Equipamentos de grande porte e expostos ao tempo, como transformadores, podem precisar de manutenção. Aliado a isso, técnicos especialistas e gestores estão muitas vezes a centenas de quilômetros de distância do local, o que inviabiliza, por exemplo, procedimentos mais acurados de manutenção preventiva e tomadas de decisão. Esse cenário vem a ser beneficiado com a solução proposta, que deve fornecer informações em tempo real frente aos requisitos do problema.

É necessária uma arquitetura de processamento para conseguir realizar essa tarefa em tempo real, entregando dados em baixa latência. O paradigma de *Fog Computing* surge nesse âmbito, definindo uma camada de processamento intermediária (névoa, ou *fog*) entre os componentes das bordas (microprocessadores, robôs, componentes IoT) e a nuvem [15, 16]. Os problemas inerentes à camada de borda, ou *edge*, como armazenamento, carga e processamento limitados, podem ser aliviados por meio da descarga de dados e tarefas a nós de processamento na *fog* [17]. Essa camada se distingue da nuvem em si pela vicinidade de seus componentes, muitas vezes em mesmo local, sendo esses roteadores, computadores, servidores e *gateways* [18].

Ao trazer o conceito geral de camadas de processamento em *fog* e *cloud* para o campo da robótica, define-se os termos *Fog Robotics* (Robótica em Névoa) (FR) e *Cloud Robotics* (Robótica em Nuvem) (CR) [19]. O conceito de FR é utilizado a fim de contornar os problemas existentes em CR: alta latência e baixa confiabilidade da rede. Em FR, é possível descarregar tarefas mais custosas a um nó de processamento mais robusto, em uma rede local de maior confiabilidade e segurança, com uma latência consideravelmente menor [20]. Dessa forma, o processamento é distribuído em todos os entes da solução, de forma a otimizar a tarefa e mitigar custos de rede, entregando o resultado final em tempo real ao usuário. Por contar com uma arquitetura distribuída em rede, é possível escalar a solução em diversos nós de processamento em ambas as camadas, algo muito necessário em ambientes industriais de grande porte. Essa proposta de arquitetura é atualmente reforçada com a presença de redes de comunicação 5G, a qual já vem trazendo benefícios em largura de banda e latência mínima a aplicações em diversas áreas, como medicina, robótica móvel e veículos autônomos [21, 22, 23]. Com isso em mente, o trabalho buscará projetar a solução proposta em um cenário de rede 5G e suas propriedades, levando em conta os requisitos para inspeção completa no cenário real de aplicação apresentado, o que vem a solucionar o gargalo da capacidade de rede com a tecnologia utilizada atualmente.

Esse trabalho busca inserir os benefícios de FR, em uma arquitetura *edge-fog*, na reconstrução 3D de ambientes de subestação. Dessa forma, o processamento mais

simples poderá ser realizado pelos componentes da camada de *edge*, que descarregarão dados e tarefas mais exigentes a um nó *fog*. Foi proposto um robô munido de sensores e processador embarcado para a aquisição dos dados em 3D e RGB, que atua como nó em *edge*. Fazendo uso de um computador como nó em *fog*, toda comprovação das vantagens da arquitetura é apresentada, bem como o estudo de escalabilidade em ambiente real, para até 5 robôs funcionando em paralelo.

Mesmo com uma arquitetura distribuída e otimizada, muitos algoritmos clássicos para processamento de imagens e dados 3D são custosos computacionalmente, prejudicando uma experiência em tempo real do usuário. Muitas soluções para esse problema demandam o uso de GPU dedicada, a qual não é presente em computadores comuns [24]. Com isso em mente, outro ponto de contribuição deste trabalho diz respeito à técnica aplicada de paralelismo do processamento de dados 3D, mais especificamente nuvens de pontos, tida como um Algoritmo de Octree Iterativa, inspirado no trabalho de Balta [25]. Este algoritmo será detalhado em suas duas versões, desenvolvidas ao longo do trabalho, destacando as melhorias incrementais entre ambas. Resultados com dados reais mostram a reduções em tempo de processamento próximos a 60% para algoritmos propostos, como a estimação de normais dos pontos 3D.

Para apresentação da solução de forma lógica e detalhada, este documento está exposto da seguinte forma:

- O Capítulo 2 apresenta a evolução deste tema de pesquisa, em especial relacionados a *Fog Computing*, bem como trabalhos relacionados atuais nas áreas de monitoramento remoto voltado a subestações de energia e processamento de nuvens de pontos e imagem em paralelo. São descritos trabalhos atuais com aplicações e descrições das características da rede de comunicação 5G, tanto práticos quanto teóricos. Por fim, também estão destacadas referências na literatura sobre processamento paralelo de imagens e nuvens de pontos, um ponto também explorado neste trabalho.

- O Capítulo 3 define os diversos algoritmos empregados para o processamento de nuvens de pontos. São detalhados todos os filtros considerados necessários, a estimação de normais e o registro de nuvens de pontos. O Capítulo também apresenta o Algoritmo de Octree Iterativa para processamento paralelo introduzido por este trabalho. No âmbito de arquitetura, são detalhadas as principais formas de dispor uma solução em rede, e a justificativa da escolha da arquitetura proposta. Tendo isso em vista, é possível alocar os processos descritos em ambas

as camadas *edge* e *fog*, descrevendo o fluxograma de funcionamento de cada uma em função dos algoritmos propostos. Como conclusão, é apresentado o robô descrito para aquisição de dados de sensores e funcionamento como nó em *edge*, em ambas as versões prototipadas.

- Os resultados são apresentados no Capítulo 4, para a versão mais recente do robô. Serão apresentados aqui os resultados do Algoritmo de Octree Iterativa e sua versão com cálculo de centroides. A arquitetura proposta é comparada com a aplicação baseada em *edge*, solução comumente apresentada em reconstruções 3D. Dados são levantados para comportamento da rede (latência e *throughput*) e de cada nó de processamento, em termos de esforço computacional, memória e armazenamento. A escalabilidade da solução fornecida pela arquitetura é apresentada como conclusão, com o funcionamento de até 5 robôs sobre dados reais em uma subestação de energia. Com base nesses resultados e na realidade do ambiente real inspecionado, é apresentado ao final um estudo e previsão para a escalabilidade da solução em um cenário com rede de comunicação 5G privada.

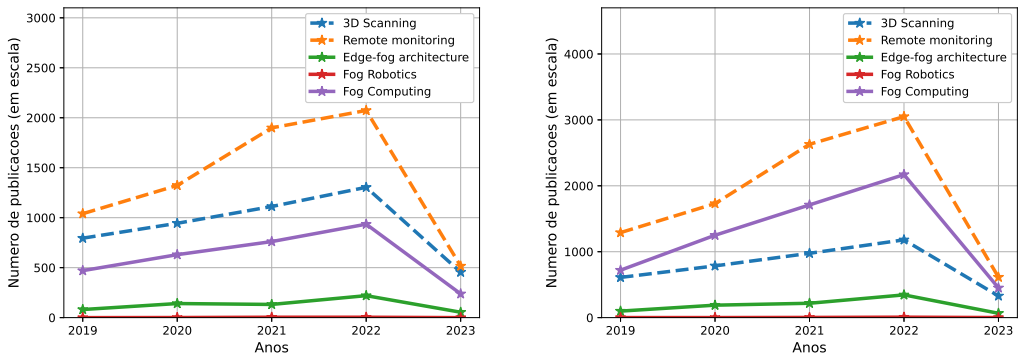
- Por fim, o Capítulo 5 traz conclusões obtidas com o trabalho, trabalhos produzidos no âmbito dessa tese de doutorado, e as principais pontos de fronteira entre realizados e previstos para trabalhos futuros.

É reforçado ao leitor que, ao longo deste trabalho, os termos "borda" e "névoa" serão constantemente escritos como *edge* e *fog*, por serem análogos em conceito, para tornar a leitura menos cansativa. Os mesmos serão utilizados tanto para indicar as camadas em si quanto para cada nó presente nas mesmas, a depender do contexto apresentado.

2 TRABALHOS RELACIONADOS

2.1 EVOLUÇÃO DOS TEMAS DE PESQUISA

Os gráficos da Figura 1a-b demonstram a evolução do interesse nos principais temas deste trabalho, em uma busca pelas palavras-chave na base de dados do *Science Direct* e *Scopus*. O valor máximo de referência para cada palavra é apresentado na Tabela 1.



(a) (b)

Figura 1: Evolução das publicações envolvendo as palavras-chave do trabalho. (a) Base *Science Direct*; (b) Base *Scopus*.

Tabela 1: Pico de publicações para as *keywords* nas bases *Science Direct* e *Scopus*.

Keywords	***Science Direct***	***Scopus***
3D Scanning	1304	1180
Remote monitoring	2073	3050
Edge-fog architecture	220	345
Fog robotics	4	7
Fog computing	935	2170

As curvas pontilhadas tratam de temas mais abrangentes na literatura, envolvendo outras áreas do conhecimento. As curvas mais sólidas dizem respeito a áreas mais

específicas da engenharia, robótica e ciência da computação. O termo *Fog computing* está presente na busca para embasar o fato de que *Fog robotics* tem já uma forte base no âmbito da computação de modo geral, para agora começar a abranger especificamente a área da robótica.

2.2 MONITORAMENTO REMOTO E REPRODUÇÃO 3D DE SUBESTAÇÕES

Há trabalhos na literatura mencionando o uso de técnicas de reconstrução para monitoramento e interação em geral com subestações. O trabalho de [8] traz uma abordagem de reconstrução de uma subestação utilizando um LiDAR montado em um veículo terrestre. A pesquisa propõe a utilização de um pipeline para processamento, desde a aquisição da nuvem de pontos e processo de filtragem, até a criação da malha e texturização. Por mais que o intuito seja fornecer um método de digitalização da subestação para operação remota e em tempo real, alguns métodos propostos no trabalho para digitalização requerem intervenção humana de um especialista. Não há também descrição de processamento *online* ou *offline*, ou mais dados nesse sentido para comprovar o uso em tempo real da solução. Por outro lado, Brosinsky [5] apresenta uma nova metodologia para replicação de centros de controle de subestações em tempo real por meio de *digital-twins*. O autor aborda o uso de unidades de medida em fasores sincronizadas temporalmente. Unindo tecnologias da época em termos de *hardware* e *software*, é proposto um centro de controle remoto para replicar a subestação e seus *softwares* SCADA, porém o conceito não chega a descrever um método para reconstituição física dos equipamentos em tempo real.

Guo [26] apresenta uma solução para monitoramento de subestação a partir da reconstituição de seus elementos de forma orientada a objeto. De forma geral, assim como em linguagem de programação, os modelos 3D dos componentes da subestação são referenciados como objetos, sendo que uma classe de equipamentos possui suas propriedades e funções para definir sua estrutura e obter dados de monitoramento. Como resultado, uma planta é reproduzida em 3D, com o monitoramento ocorrendo através da rede em tempo real. Apesar de apresentar ótimos resultados de operação e monitoramento de componentes, todo o modelo 3D é feito em CAD, não representando assim a estrutura real da subestação durante a inspeção e monitoramento remoto.

Uma situação de monitoramento e estudo remoto também é retratada em [14], onde os autores realizam uma inspeção em busca de focos de radiação na usina de Fukushima,

após situação de desastre causado pelo tsunami em 2011. Após uma ampla discussão sobre as técnicas de reconstrução 3D de ambientes, dentre elas uso de câmeras RGB e *Structure from Motion* (SfM), LiDAR 3D, e câmeras estéreo e RGBD, os autores desenvolveram um robô rastejante utilizando uma câmera RGB e outra própria para captura de raios gamma, a fim de detectar a radiação presente em pontos da subestação. Como resultados, são apresentados modelos 3D obtidos por fotogrametria de diversas imagens RGB, e a superposição dos dados obtidos de radiação gamma, conseguindo assim localizar o foco da radiação no ambiente. Apesar de apresentar uma missão bem sucedida e conseguir identificar pontos de foco de radiação, a metodologia necessita de forte processamento *offline*, sendo impraticável em tempo real para o monitoramento da subestação. Os autores concluem que, em trabalhos futuros, estão buscando uma solução com LiDAR 3D, para sanar este e outros problemas inerentes à técnica de SfM.

Finalmente, Cheng *et. al.* [13] propôs um robô para inspeção automática nomeado *LongSword*. Seu objetivo é o escaneamento de painéis de controle em subestações de energia, para detecção automática de situações de perigo e análise em tempo real do funcionamento como um todo. Montado em uma estrutura deslizante, o robô utiliza câmeras e inteligência artificial para identificar mostradores e gauges de forma autônoma, e reporta as análises para um sistema central de tomada de decisão. Na época de escrita, o robô estava em aplicação em mais de 160 salas de controle em subestações na China, se provando uma abordagem promissora para as necessidades de automação atuais. Ao mesmo tempo, não é descrita neste trabalho uma solução para monitoramento dos equipamentos de pátio, como transformadores e isoladores, bem como qualquer solução para reprodução 3D e identificação de possíveis defeitos em tal ambiente.

2.3 PROCESSAMENTO PARALELO DE IMAGENS E NUVENS DE PONTOS

É notório que o processamento de imagens pode ser uma tarefa árdua para os processadores, eventualmente demandando o uso de GPU, principalmente no contexto de reconstrução 3D por fotogrametria. O processo e otimização de nuvens de pontos, feito tanto por imagens quanto LiDAR 3D ou outras fontes, também requer cálculos relevantes, principalmente se almejado tempo real [27]. Otimizações são encontradas em técnicas de filtragem de ruídos, cálculo de normais e de *mesh* [24].

Cao [28] propõe um algoritmo para extrair e comparar descritores de imagens em paralelo no processo de reconstrução por SfM. Toda a metodologia de organização de

threads é apresentada em GPU, também removendo *outliers* das *matches* obtidas. Os resultados em bancos de dados consagrados é relevante principalmente em tempo de processamento e eficiência quando comparado a outras técnicas da academia.

Visando a otimização do cálculo de normais em dados do mundo real, [29] propõe um grafo *K-Nearest Neighbors* (kNN) sobre a nuvem de pontos completa. Dessa forma, é possível paralelizar o processamento de semelhanças entre as diversas superfícies por meio da energia calculada no grafo. Além de uma distribuição de normais mais suave ao longo da superfície, novamente o fator mais relevante nos resultados foi o tempo de processamento em comparação com técnicas clássicas e seriais, reduzido em até duas ordens de magnitude.

Por fim, Balta [25] apresenta uma abordagem de processamento paralelo utilizando *kd-trees* para segregar a nuvem de pontos em regiões menores. Cada região sofre filtragem contra *outliers* de forma paralela, o que reduz o tempo em aproximadamente metade nos resultados finais. O algoritmo, nomeado *Fast Statistical Outlier Removal*, aplica a técnica de *Statistical Outlier Removal* (SOR) de forma paralela em nuvens de pontos obtidas por LiDAR em ambiente externo. Todavia, não há menção de resultados em tempo real. Também não há menção de outros algoritmos, como a própria estimação de normais, que também podem ser realizados de forma paralela.

2.4 ARQUITETURA EDGE-FOG, FOG COMPUTING E FOG ROBOTICS, E SUA INSERÇÃO NO CONTEXTO 5G

2.4.1 DESCRIÇÃO DOS MODELOS DE ARQUITETURA EXISTENTES

Como discutido nos capítulos 1 e 2, atualmente os processos computacionais podem se dividir em três camadas mais relevantes: borda, névoa ou nuvem, e a partir daí suas derivações. Utilizar uma boa combinação dessas possibilidades resulta em mais eficiência computacional e econômica para a solução.

Aplicações em nuvem normalmente requerem um poder computacional elevado, e por vezes o uso de GPU. Exemplos de aplicações são *deep learning* e *machine learning*. Essas abordagens não devem ter sensibilidade a latência e confiabilidade da rede, que são os pontos fracos da abordagem em nuvem [19]. Além do mais, os dados da aplicação são enviados para a internet, potencialmente reduzindo a privacidade e segurança da aplicação. Tendo isso em vista, o trabalho buscou utilizar recursos locais para realização da solução de escaneamento.

Aplicações robóticas tendem a depender de processamento na borda, devido à leitura em tempo real de sensores e controle de baixo nível de atuadores. Muitas vezes todo o processamento é realizado na borda, que possui restrições como capacidade de processamento e armazenamento limitadas, e menor disponibilidade energética e de espaço/carga. Tentar contornar esse problema pode encarecer e até mesmo alterar a solução. No caso de escaneamento, soluções deste tipo tendem a obter dados para o pós processamento e análise *offline*, ou transmitir dados crus para monitoramento por parte do usuário. Esse tipo de arquitetura está ilustrado na Figura 2a.

Como visto nas aplicações em névoa, servidores ou nós centrais de processamento são responsáveis por receber dados obtidos pelos sensores e dispositivos IoT nas bordas e processá-los [30, 31]. Porém, o repasse de dados não tratados ou filtrados por parte da borda tende a consumir maior largura de banda da rede, o que reduz a escalabilidade e aumenta a latência da solução (Figura 2b).

Tendo isso em vista, uma arquitetura inteligente, utilizando todos os agentes locais disponíveis, seria a arquitetura *edge-fog*. Dessa forma, parte do processamento pode ser feito em *edge*: a comunicação rápida com sensores IoT, pré-processamentos a fim de aliviar a transmissão em rede, e utilização de forma eficiente do poder computacional menor. Os dados são descarregados para o nó de processamento em *fog*, o qual possui maior capacidade computacional, energética e de armazenamento, em uma comunicação em rede local, o que resulta em baixa latência e menor tempo de processamento [19, 32]. O resultado de uma boa distribuição pode otimizar consideravelmente os dados quando comparados a uma arquitetura baseada em *edge*, como será mostrado nos resultados do Capítulo 4. Como consequência, mais nós podem ser adicionados à borda, fornecendo escalabilidade à solução. A medida que se faz necessário, o mesmo pode ocorrer com a camada em névoa. Essa arquitetura é representada na Figura 2c.

2.4.2 FOG COMPUTING E ARQUITETURA EDGE-FOG NA LITERATURA

O conceito de uma arquitetura edge-fog, que também é por vezes denominada *fog computing*, ou computação em névoa, varia em detalhes na literatura. Em termos gerais, como definido pelo *National Institute of Standards and Technology* em [33], a computação em névoa existe próxima a dispositivos em *edge*. Ela tem como objetivo dar suporte a operações dependentes de baixa latência, que devem ser verticalizadas e em camadas, permitindo sua escalabilidade, com processamento e armazenamento distribuídos nos elementos da rede.

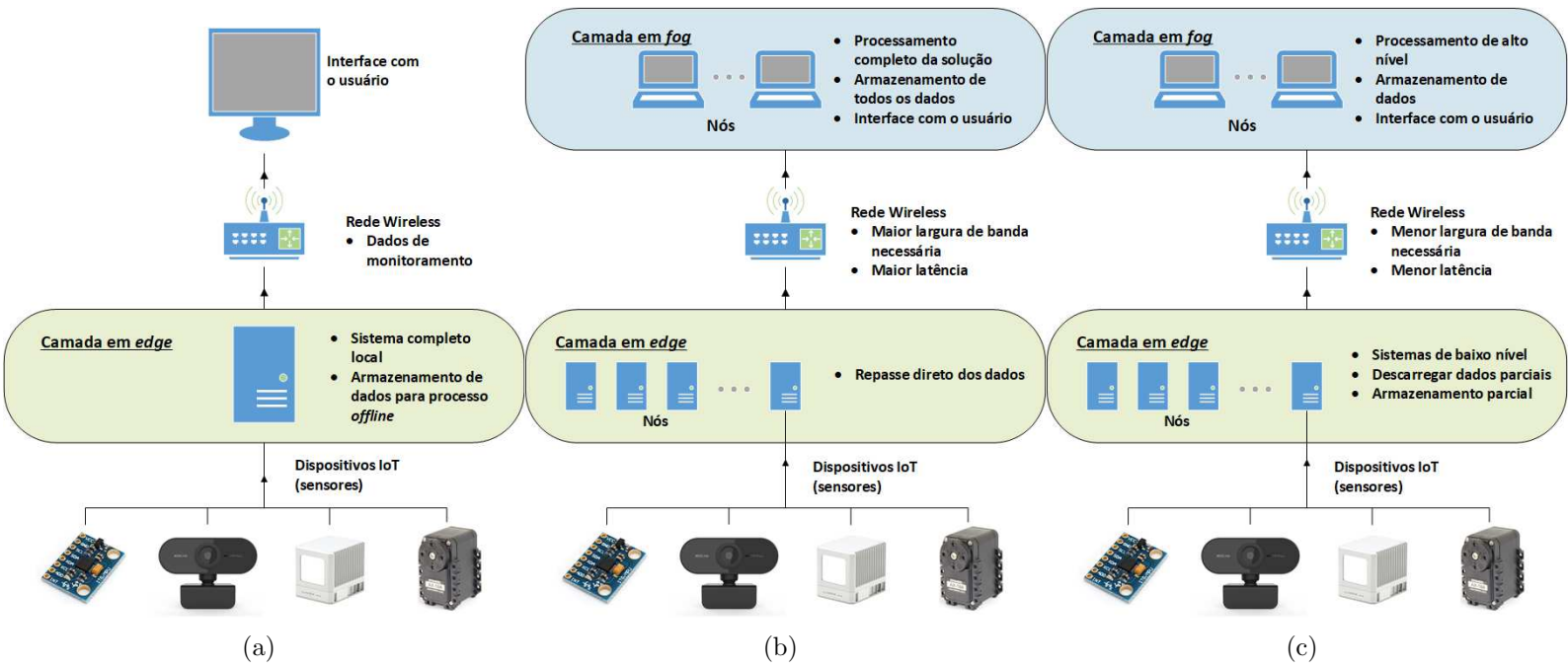


Figura 2: Possibilidades de arquitetura (a) Baseada em *edge*, ou borda; (b) Camada em *edge* repassando diretamente dados para camada em *fog* processar; (c) Arquitetura *edge-fog*, com processamento em ambas as camadas, descarregamento de dados, rede eficiente e interface com o usuário.

Um paralelo entre a computação em névoa e a área da robótica é traçado por Gudi *et.al.* [19], introduzindo o conceito de FR. Pela necessidade de processamento de algoritmos especiais, envolvendo *deep learning* e *machine learning*, robôs necessitam de maior poder computacional e uma latência que permita uma aplicação de quase tempo real, que seria o maior problema quando a arquitetura escolhida se baseia em CR. Os autores apresentam um debate sobre aplicações críticas em termos de latência, como um robô de combate a incêndio, e mostram resultados comparativos entre um servidor em névoa e em nuvem para aplicações específicas. É concluído que o uso de processamento em névoa reduz latência e custos para montagem do robô em si, também garantindo escalabilidade de solução, porém resultados quantitativos nesse sentido não são apresentados. O mesmo foco pode ser visto em [20], onde os autores enfatizam a evolução e contribuição de CR para a área, mas apontam a necessidade de uma menor latência e dependência da rede. Destaca-se a aplicabilidade de FR em robôs com operações de tomada de decisões rápidas.

Uma aplicação de FR é descrita em [30]. Neste trabalho, o robô IGOR foi desenvolvido para manipulação de caixas de forma automática e/ou assistida. O trabalho descreve o paradigma de processamento em todos os níveis computacionais, sendo estes *edge, fog* e *cloud*, despendendo atividades mais críticas e imediatas de controle na

borda, e a medida que a necessidade de inteligência e capacidade computacional aumentam, os algoritmos são dispostos em servidores na *fog* (um *smartphone*) ou *cloud*. Toda a solução é apresentada, desde algoritmos básicos de controle a dados de sincronização e latência. Como conclusão, por mais que as taxas de acerto quando em operação assistida fossem consideradas satisfatórias, a latência apresentada quando assistido pela nuvem é tão relevante que causou uma redução drástica em comparação com a aplicação em névoa.

Também no contexto de *Simultaneous Localization and Mapping* (Localização e Mapeamento Simultâneos) (SLAM), o trabalho de Sarker *et. al.* [34] propõe uma solução de processamento em *edge, fog* e *cloud* para processamento de dados de um robô móvel de inspeção em ambientes industriais. É mostrado que dados de imagem e vídeo tendem a elevar a latência da rede de forma relevante quando há um aumento no número de robôs funcionando em paralelo. Portanto, toda parte crítica computacionalmente é trabalhada em *fog*, deixando a nuvem responsável por interações com o usuário e armazenamentos de acesso menos crítico principalmente. Tarefas de controle ainda seriam performadas em *edge*, dentro do robô. Como experimento, um protótipo de veículo utilizando LiDAR 2D envia dados para o processamento na *fog*, avaliando o sistema em termos de gasto energético em *edge*. As conclusões demonstram a economia de energia devido ao processamento reduzido, mas não estudam aspectos da rede ou eficiência de cada etapa do processamento sendo realizada em mais de uma camada de rede.

Em [35], os autores mencionam o advento de utilização da FR para aplicações envolvendo *deep learning*. É mencionado como computação em névoa pode realizar tarefas como filtragem, reconhecimento, pré-processamento e inferência, reduzindo a uso da banda de rede local. Além de tudo, ao manter as aplicações mais próximas do usuário, a segurança dos dados pode ser melhor controlada quando comparada a alguma aplicação em nuvem. A latência também é beneficiada pela computação em *fog*, reduzindo em 4 vezes o tempo de inferência em reconhecimento de padrões para manuseio autônomo de objetos. Esses conceitos de FR também são explorados e comprovados ao longo dessa tese, em consonância com o estado da arte observado na literatura.

2.4.3 ARQUITETURA *EDGE-FOG-CLOUD* NO CONTEXTO DE REDES DE COMUNICAÇÃO 5G

A distribuição de tarefas entre as diversas camadas de uma arquitetura envolvendo *edge*, *fog* e *cloud* é vista em [36]. Os autores apresentam como o processamento distribuído em camadas poderia ocorrer usando uma rede 5G. Eles discutiram os conceitos, desde elementos de hardware para descarregamento de dados e níveis hierárquicos de *edge*, *fog* e computação em nuvem. Como conclusão, a computação em *fog* aproxima as capacidades da *cloud* dos dispositivos IoT, diminuindo problemas de *delay*, em uma abordagem local. Eles também propõem um método para otimizar o rendimento em rede distribuindo tarefas entre as diversas camadas e nós. Outro trabalho que trata de distribuição e *offload* de processamento de imagens para SLAM é visto em [37], onde os autores demonstram qual a melhor abordagem a ser tomada a depender das características da rede, e como o cenário 5G favorece essa transferência de dados para processamento em *fog*.

Como exemplo de uma distribuição envolvendo também essas três camadas utilizando uma rede 5G, no contexto de operação de resgate, tem-se o trabalho de [38]. que propõe a rede denominada X-IoCA, descrevendo um contexto de um futuro com uma arquitetura vasta e heterogênea tanto em tipos de rede (BLE, wifi, Lora) quanto dispositivos (veículos terrestres, drones, celulares). Parte dessa heterogeneidade é também descrita nesta tese, confirmando o conceito e processamento em camadas.

Outro contexto muito visto atualmente na literatura com o emprego de uma arquitetura distribuída e as capacidades da rede 5G é o de navegação de carros autônomos, os quais fazem uso de LiDAR para reconstrução 3D visando mapeamento e navegação. Como visto no trabalho de [23], os autores usaram uma arquitetura *edge*, *fog* e *cloud* para processamento de LiDAR, leitura de câmera, dados e descarregamento de processamento para algoritmos de SLAM na camada de névoa e, finalmente, os recursos da nuvem para realizar operações de mapeamento. Eles descreveram o sistema e os processos que compõem cada camada e apresentaram o desempenho do sistema em função do número de veículos funcionando como nós na borda. Também almejando operações de navegação autônoma, [39] apresentou uma arquitetura para realizar o reconhecimento de objetos ao longo das estradas, descrevendo como cada camada é construída e planejada para lidar com o grande volume de dados esperado de fontes heterogêneas (câmeras, LiDAR, GPS, entre outros). Os resultados são relacionados ao desempenho de reconhecimento do sistema e escalabilidade de nós de borda, que é uma meta crucial nesses cenários de aplicação. Isso vai ao encontro da escalabilidade estudada e também

obtida no trabalho desta tese.

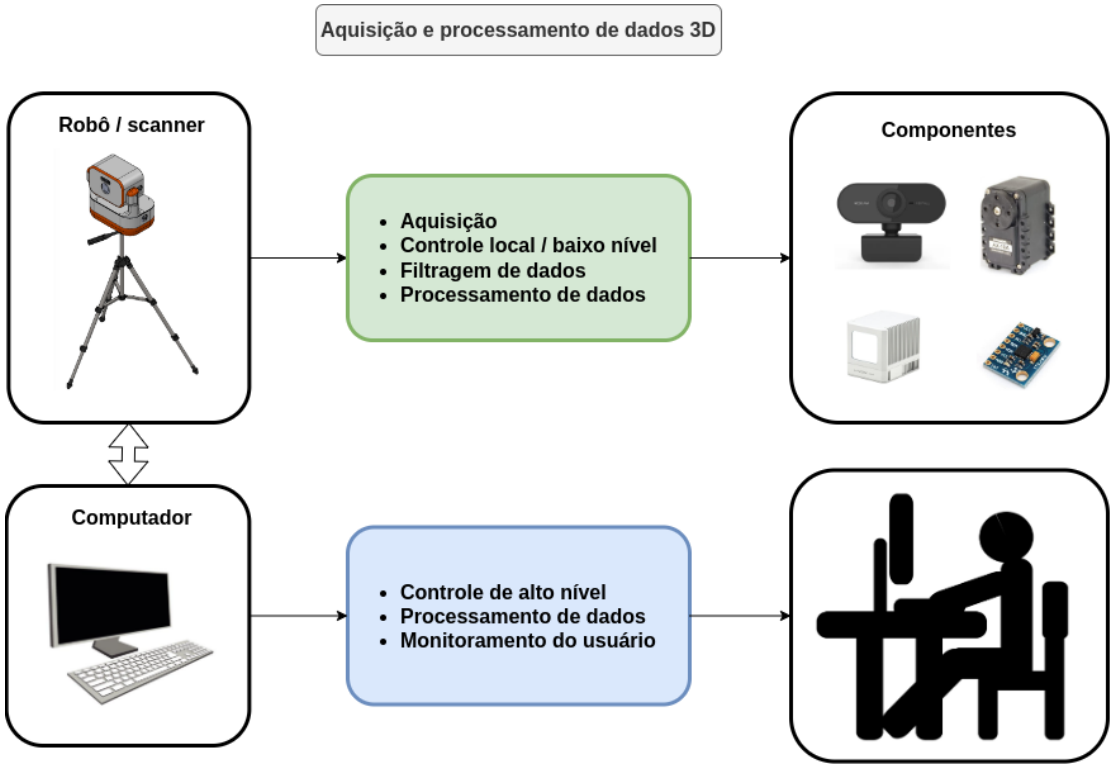
3 MATERIAIS E MÉTODOS

O fluxograma da Figura 3a ilustra a solução atual, de forma generalizada: temos a intenção de realizar o escaneamento e reprodução 3D do ambiente, e para isso precisamos de dois entes iniciais, um robô ou *scanner* que contenha os dispositivos necessários, e o computador de monitoramento e interface com o usuário, onde os resultados estarão presentes eventualmente. Cada ente pode realizar as tarefas citadas na coluna do meio nos quadrados verde e azul.

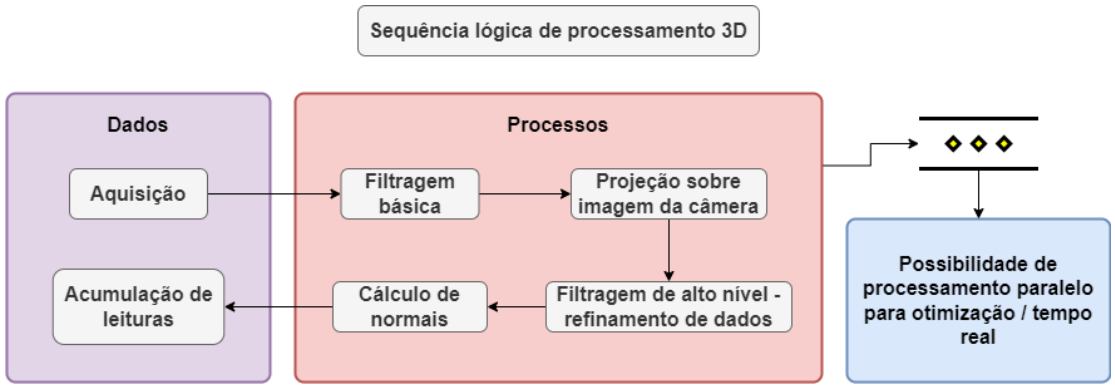
O leitor pode ter uma noção mais específica do processamento de dados vindos deste *scanner* na Figura 3b. Essa sequência deve ser realizada da melhor maneira possível a fim de obter o processamento mais eficiente e balanceado possível entre os entes apresentados anteriormente. Também é proposta a hipótese: podemos otimizar não só a alocação dos processos, mas como os mesmos são realizados em si? Isso é provavelmente o resultado obtido com o processamento paralelo, que também é um resultado deste trabalho.

Neste capítulo serão apresentados as técnicas e os materiais empregados para desenvolvimento da solução final. Nas seções seguintes serão destacados:

- Técnicas algorítmicas para processamento de nuvens de pontos, clássicas e inovadoras, incluindo pré-processamento por filtragem, divisão em octree iterativa, filtro de *raycasting* e covariância, estimação de normais e adição de pontos não vistos por buscas de vizinhos usando *kd-tree*. A sequência de apresentação reflete também uma sequência lógica de aplicação das mesmas.
- Descrição de custo computacional das tarefas almejadas, e sua consequente distribuição na arquitetura *edge-fog* proposta.
- Em paralelo, a descrição do robô desenvolvido em termos de *hardware*, *software* e matemática para execução das tarefas em *edge*.
- Fluxogramas mostrando a organização dos processos em *edge* e *fog*.



(a)



(b)

Figura 3: Fluxogramas de resumo do sistema. (a) Entidades envolvidas na aquisição, seus componentes e possíveis tarefas. (b) Sequência lógica de processamento de dados 3D, com possibilidade de otimização por algoritmos de paralelismo.

3.1 TÉCNICAS ALGORÍTMICAS

3.1.1 PREMISSAS E FILTRAGENS BÁSICAS

O LiDAR possui um *frame* coordenado definido como $\mathbb{F}_{li}$, o qual neste trabalho é transformado e definido como normalmente utilizado em câmeras. Sua origem se encontra no centro virtual do sensor, que é aproximadamente seu centro geométrico. Nele, o eixo Z aponta da origem para a região frontal do sensor, o eixo X está da

origem para a direita (olhando no sentido do eixo Z), e o eixo Y aponta para baixo do sensor. Os ângulos de rotação em torno dos eixos XYZ são θ, ψ e ϕ, respectivamente. A Figura 4 ilustra essa definição, com o sensor montado no robô.

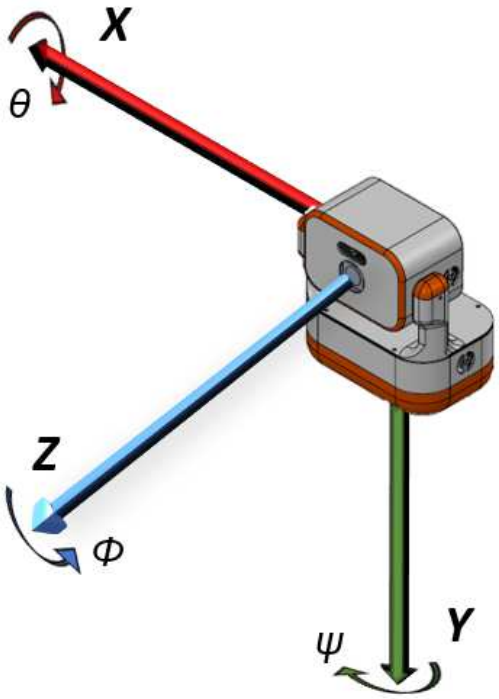


Figura 4: Orientação dos eixos no *frame* do LiDAR $\mathbb{F}_{li}$.

A primeira abordagem a ser tomada no processo de captura de nuvem de pontos é a filtragem de más leituras. A depender do *driver* de leitura adotado, leituras que não são válidas (como as de objetos muito próximos, por exemplo) são levadas à origem $\mathbf{O} \in \mathbb{R}^3$ de $\mathbb{F}_{li}$, adicionando um dado desnecessário na nuvem de pontos. Como certamente não há a possibilidade de existir pontos na origem da medição, todos esses pontos podem ser descartados.

Outro modo de filtragem é a profundidade de leitura, medida do eixo Z. Pontos muito distantes tendem a ser menos precisos e trazer informações desnecessárias para o usuário em muitas aplicações. Dessa forma, supondo um nuvem de pontos C_{li} aquisitada em $\mathbb{F}_{li}$, um limite Th_z é definido para o valor em Z_k do ponto $\mathbf{P}_k \in \mathbb{R}^3, \forall C_{li}$, e se ultrapassado, o ponto é desconsiderado. A Equação 3.1 define ambas as regras básicas de filtragem descritas.

$$\mathbf{P}_k = \begin{cases} \text{bom, se } \mathbf{P}_k \neq \mathbf{O} \cap Z_k < Th_z \\ \text{descartado, senão} \end{cases}, \forall \mathbf{P}_k \in C_{li} \tag{3.1}$$

3.1.2 PROJEÇÃO DE PONTOS 3D SOBRE IMAGEM 2D

De posse do sensor LiDAR e de uma câmera RGB, a partir de calibração extrínseca e intrínseca de ambos, e projeções matemáticas, é possível fornecer cor de forma precisa à nuvem de pontos obtida pelo primeiro. A câmera possui *frame* coordenado no espaço 3D bem como o LiDAR, definido como $\mathbb{F}_{cam}$, com os eixos de forma semelhante aos vistos na Figura 4, e origem no sensor ótico.

Suponha a nuvem de pontos C_{li} obtida pelo LiDAR, e $\mathbf{P}_{li_k} \in C_{li}$ sendo os pontos desta nuvem em coordenadas homogêneas ($\mathbf{P}_{li_k} \in \mathbf{R}^4$), como ilustrado na Figura 5.

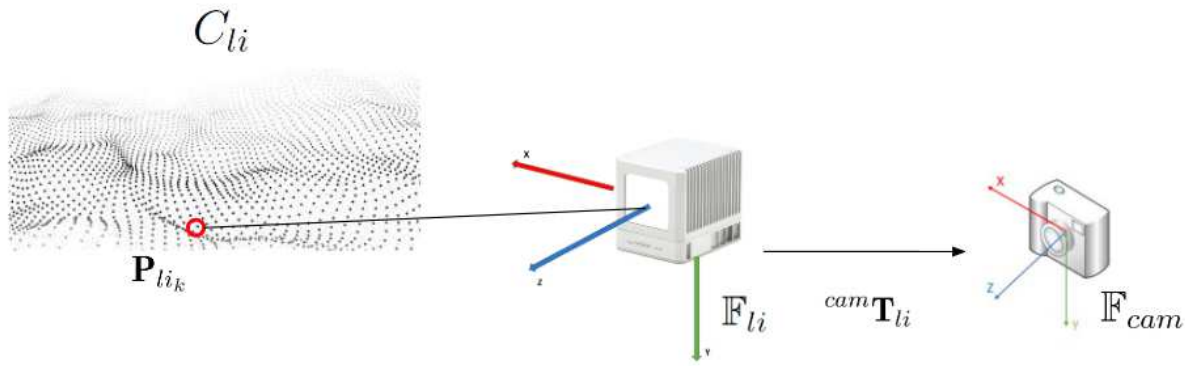


Figura 5: Projeção dos pontos 3D do LiDAR do *frame* $\mathbb{F}_{li}$ para o *frame* $\mathbb{F}_{cam}$.

A imagem RGB obtida pela câmera é definida por I_t, onde cada pixel $\mathbf{p}_k$ possui um valor para cada canal, ou seja, $\mathbf{p}_k \subset [0,2^8]^3, \forall \mathbf{p}_k \in I_t$. Dessa forma, a Equação 3.2 projeta os pontos de C_{li} na imagem I_t, a fim de atribuir a cor do pixel projetado ao ponto em 3D. A Equação 3.3 define a transformação dos pontos do *frame* $\mathbb{F}_{li}$ para $\mathbb{F}_{cam}$, com origem na câmera, obtida por calibração. Por fim, a Equação 3.4 define a matriz intrínseca da câmera $\mathbf{K}_{cam} \in \mathbb{R}^{3\times3}$.

$$\mathbf{p}_{h_k} = \mathbf{K}_{cam} \cdot {}^{cam}\mathbf{T}_{li} \cdot \mathbf{P}_{li_k}, \forall \mathbf{P}_{li_k} \in C_{li} \tag{3.2}$$

$${}^{cam}\mathbf{T}_{li} = \begin{bmatrix} \mathbf{R}_{cam}^{3\times3} & \mathbf{t}_{cam}^{3x1} \end{bmatrix} \tag{3.3}$$

$$\mathbf{K}_{cam} = \begin{bmatrix} f_x & \alpha_y & c_x \\ \alpha_x & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

onde $\mathbf{p}_{h_k} \in \mathbb{R}^3$ é o pixel projetado em coordenadas homogêneas; $\mathbf{R}_{cam}^{3\times3} \in \mathbb{SO}_3$ (conjunto das matrizes ortogonais especiais) e $\mathbf{t}_{cam} \in \mathbb{R}^{3x1}$ são a matriz de rotação e vetor de translação de $\mathbb{F}_{li}$ para $\mathbb{F}_{cam}$, respectivamente; e f_i e c_i são os focos da câmera e coordenadas do eixo ótico na imagem, ou centro, em X e Y.

Ao final, para obter as coordenadas reais do pixel projetado $\mathbf{p}_k \in \mathbb{Z}^2$, e então sua cor, é necessário dividir o valor de $\mathbf{p}_{h_k} = \begin{bmatrix} u & v & w \end{bmatrix}^T$ pela sua última coordenada, como definido na Equação 3.5.

$$\mathbf{p}_k = \begin{bmatrix} u/w \\ v/w \end{bmatrix} \tag{3.5}$$

3.1.3 ALGORITMO DE OCTREE ITERATIVA

A divisão espacial em octree é uma técnica consagrada na computação, principalmente para busca em espaços 3D de forma mais eficiente. A mesma estabelece uma avaliação contínua a partir de divisão de um espaço maior (um nó) em oito partes menores - as folhas. Em sequência, cada folha pode ser considerada um novo nó, e ser dividida novamente [40], o que cria um estrutura de divisão em árvore. A profundidade da octree depende de quantas vezes a operação de divisão se repetiu. Como proposto em [25], seu uso pode ser empregado em processo paralelo de nuvens de pontos, dobrando a velocidade de processamento.

Supondo uma nuvem de pontos capturada e colorida C_{cam}, é possível dividi-la em 8 folhas $C_{cam_k}, [1 \leq k \leq 8]$, e assim por diante em cada folha, a depender da profundidade desejada pelo usuário. O ponto principal, para dados reais, é a baixa homogeneidade na distribuição de pontos no espaço. Dessa forma, uma dessas folhas pode conter de nenhum a praticamente todos os pontos da nuvem do nó original. A solução proposta nesse trabalho consiste então em uma octree iterativa, de tal forma que folhas que contenham ainda um valor expressivo de pontos continuem sofrendo divisões até que todas obtenham uma determinada quantidade de pontos. Note que a divisão ocorrerá somente nas folhas que necessitarem, ocorrendo assim uma profundidade variada ao longo da árvore. Dessa forma, regiões de alta densidade são divididas e não se tornam

gargalos. Além de tudo, regiões com pouca quantidade relativa de pontos podem ser descartadas no processo, filtrando assim a saída e sequência de processamento.

Duas metodologias foram propostas para a divisão das folhas, o que resultou em duas versões do algoritmo. Inicialmente, o conceito se baseava na divisão geométrica simétrica da folha densa original, o que requer menos cálculos, mas não valoriza a distribuição de densidade da nuvem de pontos propriamente dita. A partir disso, a segunda metodologia se resume em subdividir a folha densa a partir do centro de massa da nuvem de pontos, ou centroide, obtido de forma semelhante ao que será apresentado na Equação 3.8 da subseção seguinte. Assim, mesmo requerendo mais operações para determinar o centro de divisão, temos uma busca guiada sobre a geometria da nuvem de pontos, o que tende a requerer menos iterações e possivelmente menos tempo, com nuvens de saída mais homogêneas. A Figura 6 ilustra o Algoritmo de Octree Iterativa sobre uma nuvem de pontos de entrada, e os resultados para ambas as formas de cálculo de divisão de folhas serão apresentadas no Capítulo 4.

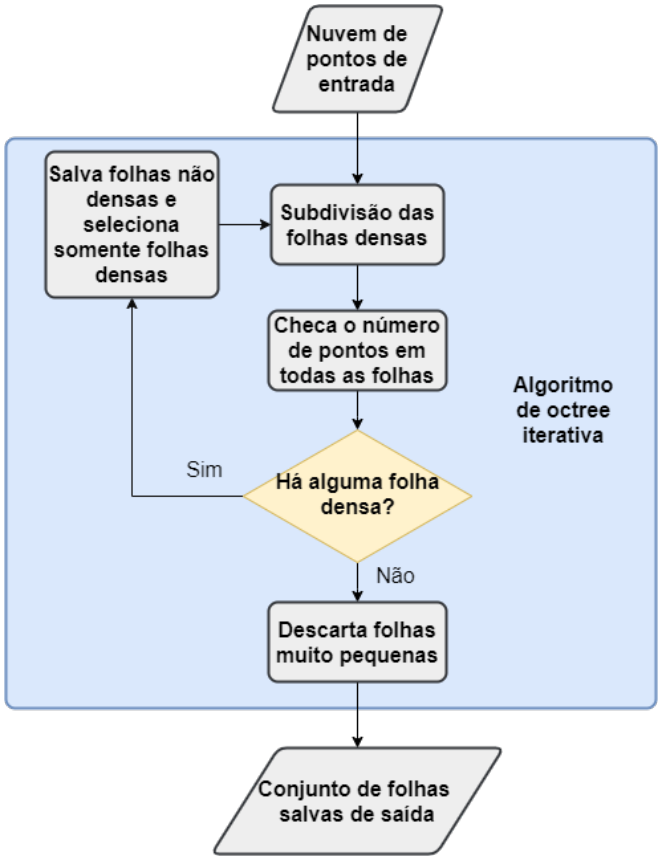


Figura 6: Fluxograma do Algoritmo de Octree Iterativa.

3.1.4 FILTROS RAY CASTING, SOR E COVARIÂNCIA

Os filtros de *ray casting*, inspirados pelo trabalho de [41], e covariância, surgem da necessidade de filtragem de ruídos encontrados nas bordas de transição entre objetos e fundo da cena, e oclusão entre os mesmos ou entre objetos, principalmente em ambientes internos [42, 43]. Esses ruídos ocorrem principalmente na direção radial de aquisição.

Para o filtro de *raycasting*, considere uma linha saindo da origem de $\mathbb{F}_{li}$ em uma direção definida pelo vetor $\mathbf{d}_{rc} \in \mathbb{R}^3$, obtida através das coordenadas esféricas de latitude e longitude lat e lon, respectivamente (Equação 3.6). Se essa direção tiver uma interseção com a nuvem de pontos C_{cam} no ponto $\mathbf{P}_{cam_k} \in C_{cam}$, este ponto é considerado como uma leitura verdadeira. Pontos que não sofreram interseção devem estar oclusos ou mais afastados, indicando um possível ruído radial.

$$\mathbf{d}_{rc} = \begin{bmatrix} \operatorname{sen} lat \cdot \operatorname{sen} lon \\ \cos lat \\ \operatorname{sen} lat \cdot \cos lon \end{bmatrix} \tag{3.6}$$

O filtro de covariância é aplicado de forma paralela num contexto mais local da nuvem de pontos, na vizinhança de cada ponto $\mathbf{P}_{cam_k} \in C_{cam}$. Essa é definida como uma fração da nuvem inicial $C^*_{cam_k} \subset C_{cam}$ contendo N pontos. Essa nuvem é rotacionada de tal forma que o eixo Z coincida com a direção radial estabelecida entre a origem e $\mathbf{P}_{cam_k}$. Os ângulos de rotação ψ^* e θ^* de *pan* e *tilt* dessa operação são definidos na Equação 3.7, a partir das coordenadas cartesianas X_k, Y_k e Z_k do ponto $\mathbf{P}_{cam_k}$.

$$\psi^* = \arctan(Y_k, Z_k) \quad \theta^* = \arctan(X_k, Z_k) \tag{3.7}$$

Após a rotação, o centroide $\mathbf{c}^*_k = \begin{bmatrix} \overline{X} & \overline{Y} & \overline{Z} \end{bmatrix}^T$ e a matriz de covariância $\mathbf{\Sigma}^*_k$ são calculados como nas Equações 3.8 e 3.9 com base nos $N*$ pontos de $C^*_{cam_k}$. Se os termos $\mathbf{\Sigma}^*_{k_{(1,1)}}$ e $\mathbf{\Sigma}^*_{k_{(2,2)}}$ (covariâncias em X e Y) são menores que $\mathbf{\Sigma}^*_{k_{(3,3)}}$ (covariância em Z) por um valor proporcional definido $n_{cov} \in \mathbb{R}$, $C^*_{cam_k}$ é considerada altamente distribuída na direção radial. Dessa forma, P_{cam_k} é removido de C_{cam}.

$$\mathbf{c}^*_k = \begin{bmatrix} \overline{X} \\ \overline{Y} \\ \overline{Z} \end{bmatrix} = \frac{1}{N* - 1} \sum_{n=1}^{N*} \mathbf{P}_n, \text{ onde } \mathbf{P}_n = \begin{bmatrix} X_n \\ Y_n \\ Z_n \end{bmatrix} \tag{3.8}$$

$$\mathbf{\Sigma}_k^* = \frac{1}{N*-1}\sum_{n=1}^{N*}\left(\mathbf{P}_n - c_k^*\right)\left(\mathbf{P}_n - c_k^*\right)^T,\ \forall \mathbf{P}_n \in C_{cam_k}^* \tag{3.9}$$

Por fim, o filtro SOR é utilizado para acusar *outliers* esporádicos na nuvem de pontos. Baseado em estatística, ele busca a média e desvio padrão da distância entre os vizinhos dos pontos na nuvem. Caso um ponto esteja consideravelmente longe de seus vizinhos segundo estes parâmetros, é descartado. Mais informações podem ser obtidas na implementação original, em [44]. Na prática, como o sensor LiDAR envia uma quantidade relativamente densa de dados, os objetos escaneados apresentam pontos com distância relativamente uniforme em sua vizinhança, a não ser quando observados ruídos. Portanto, um desvio padrão de desvio da média se mostrou um dado generalizável para as aplicações.

3.1.5 ESTIMAÇÃO DE NORMAIS

A estimação de normais segue a implementação clássica, mas processada em paralelo. A vizinhança do ponto $\mathbf{P}_{cam_k} \in C_{cam}$ é inspecionada, de tal forma que a *Principal Component Analysis* (PCA) sobre a matriz de covariância da mesma retorne autovalores e autovetores, que resulta no vetor normal. Mais detalhes estão descritos na referência [45].

O fato relevante e introduzido neste trabalho diz respeito à orientação das normais. A suavidade das orientações das normais em uma superfície é crucial para processos seguintes de reconstrução 3D, como reconstituição de *mesh* [29]. Para evitar que cálculos matemáticos resultem em normais apontadas para "dentro" de uma superfície, todas as normais são inspecionadas com o objetivo de terem o sentido apontando para a origem de $\mathbb{F}_{li}$. Suponha o processo de cálculo da normal como sendo $\mathbf{N}_k = F(\mathbf{P}_k) \in \mathbb{R}^3$, onde F simboliza a função de cálculo de normais, e $\mathbf{P}_k \in C_{li}$ é um dos pontos da nuvem em $\mathbb{F}_{li}$. A correção de sentido da normal é obtida pelo produto interno entre o vetor normal $\mathbf{N}_k$ no ponto $\mathbf{P}_k$ e o vetor direcional $\mathbf{OP}_k \in \mathbb{R}^3$, entre o mesmo ponto e a origem de $\mathbb{F}_{li}$.

$$\mathbf{N}_k = \begin{cases} F(\mathbf{P}_k), \text{ se } \mathbf{N}_k \cdot \mathbf{PO}_k > 0 \\ -F(\mathbf{P}_k), \text{ senão} \end{cases},\forall \mathbf{P}_k \in C_{li} \tag{3.10}$$

3.1.6 REGISTRO DE PONTOS NÃO VISTOS POR KD-TREE

Ao registrar duas nuvens de pontos aquisitadas em poses diferentes, é possível acumular pontos repetidos, uma vez que haja interseção de cenário entre as aquisições. É de interesse que essa interseção ocorra, tanto para capturar detalhes de pontos de vista diferentes quanto para otimizar o registro em si. Porém, pontos repetidos geram um gasto de memória e um processamento ineficiente. Portanto, ter conhecimento de pontos em ambas as nuvens que estariam representando a mesma entidade na cena é importante, e sua filtragem reduz drasticamente a memória de RAM e de armazenamento, bem como o custo computacional.

Suponha duas nuvens de pontos, sendo elas C_i e C_j obtidas nas capturas i e j, respectivamente, sendo $j > i$. Assim, para cada ponto $\mathbf{P}_{k_j} \in C_j$, suas coordenadas são buscadas por kd-tree em C_i. Se a busca encontrar algum ponto em um raio pré-determinado R_n, $\mathbf{P}_{k_j}$ é considerado repetido, e portanto descartado de C_j. O resultado da nuvem filtrada é definido por C_j^*.

Como conclusão deste processo, ambas as nuvens podem ser registradas, sendo o resultado final a união de ambos conjuntos de pontos R_{ij} definida na Equação 3.11.

$$R_{ij} = C_i \bigcup C_j^* \tag{3.11}$$

3.2 ARQUITETURA EDGE-FOG

3.2.1 PREMISSAS DE DESENVOLVIMENTO

No decorrer da distribuição da solução em arquitetura *edge-fog*, as premissas que seguem foram adotadas para obter a melhor distribuição de processos e utilização da rede. Elas serão esclarecidas no decorrer desta seção.

- O tempo de processamento de cada etapa foi tomado em PU, para que o tempo possa ser estimado em segundos em ambos os processadores da borda e névoa.
- Necessidades de transferência de dados também foram medidas para minimizar o *throughput* na rede.
- Todo o processo é concebido a princípio como embarcado na borda, até que sua capacidade seja atingida e comece a haver um gargalo em termos de duração do processamento.

- Buscou-se evitar o trânsito de imagens em HD pela rede, por consumir larga banda e denegrir o monitoramento em tempo real.
- Além de evitar um gargalo na borda, buscou-se descarregar o máximo possível de processamento na névoa para aproveitamento de poder computacional, sem prejudicar a latência da operação.
- O avanço da solução em *edge-fog* é comparado ao final com os resultados processados de forma embarcada na borda, a fim de comprovar a eficiência da arquitetura proposta e a melhora de experiência do usuário.

3.2.2 AVALIAÇÃO DOS PROCESSOS EM PU

Para início do desenvolvimento, dados de LiDAR, *Inertial Measurement Unit* (Unidade de Medição Inercial) (IMU) para obtenção de odometria e câmera (imagem HD) foram colhidos com um *laptop* em ambientes externos. Sobre esses dados, todos os processos descritos na Seção 3.1 foram avaliados em termos ME e MS, bem como tempo de processamento.

Para explicitar melhor os procedimentos para estimação do tempo, principalmente em um ambiente em rede, é necessário citar alguns passos. Primeiramente, ambas as máquinas devem estar baseadas em uma mesma fonte de *clock*, procedimento que foi certificado para os sistemas baseados em linux. Em seguida, para estimação do tempo de execução de procedimentos, foram utilizadas bibliotecas próprias da linguagem C++, retornando precisão em microssegundos, ou, para dados de latência de mensagem, foi usada a diferença entre o tempo atual e o *timestamp* atribuído à mensagem. Para o tempo de processamento poder ser comparado entre computadores, uma unidade de medida foi definida, logo todas as medidas estão dispostas em PU. Essa unidade corresponde ao tempo para o processador realizar uma soma de dois inteiros por 100 vezes em um núcleo.

Os dados obtidos estão expostos na Tabela 2.

Tabela 2: Resultados de PU, ME, e MS (Em Megabytes) para cada processo estudado.

Linha	**Nome do processo**	**PU** ($x10^5$)	**ME (MB)**	**MS (MB)**
1	Aquisitar odometria	-	-	-
2	Aquisitar nuvem de pontos	277,80	11,310	11,310
3	Filtragem básica	2,37	5,378	4,351
4	Colorir nuvem de pontos	14,69	4,721	5,438
5	Capturar imagem HD e salvar	29.30	5.932	3.230
6	Reduzir resolução de imagem e enviar	13.80	5.932	0.370
7	Enviar imagem HD	412.71	5.932	5.932
8	Algoritmo de Octree Iterativa	9.72	2.266	2.266
9	Filtro *raycasting* e covariância	59.75	2.266	1.805
10	Filtro SOR	22,82	1.805	1,382
11	Estimação de normal	46,32	1,382	2,486
12	Adição de pontos não vistos com *kd-tree*	74,42	2,486	1,235
13	Registrar nuvem final	3,16	1,235	-

Os valores para aquisição de orientação do robô no mundo (odometria) são irrelevantes quando comparados aos demais, tendo frequência elevada em *edge*.

Como pode ser visto comparando as linhas 6 e 7 da Tabela 2, o custo de latência (através do PU) e *throughput* (através da MS) na rede diminui drasticamente com o envio de uma imagem em menor resolução, mais precisamente reduzida a um quarto da original, resultando em 420x270. Empiricamente, este valor é suficiente para o acompanhamento em baixa latência da operação por parte do usuário. Portanto, o único processo da tabela que não é alocado na solução é o envio de imagem em HD (linha 7). Todos os outros processos apresentam um PU inferior inclusive ao de aquisição da nuvem de pontos e de ordem de grandeza semelhantes. Além disso, agregam valor à qualidade da nuvem de pontos e reconstrução final.

3.2.3 ALOCAÇÃO DOS PROCESSOS E JUSTIFICATIVAS

O total de processamento em termos de PU é de $554{,}15x10^{-5}$ dispensando a linha 7 da Tabela 2. Os seguintes fatos são levados em conta neste momento:

- Deste valor, 50,13% é referente ao tempo de aquisição da nuvem de pontos para uma captura fiel da cena, e deve ser performada em alta velocidade na borda,

onde estará conectado o LiDAR.

- Tarefas de aquisição de imagens também devem ser realizadas na borda, por razões semelhantes.

- O armazenamento da imagem em HD também será realizado na borda, para poder poupar banda da rede e reduzir latência, como já comprovado.

- Filtragens básicas e coloração de nuvem (linhas 3 e 4) tomam somente 3% do total em PU. A primeira performa uma redução de 19,1% em termos de memória com ruídos, o que poupa banda de rede. Porém, o maior impacto se dá pelo fato de a coloração ter um resultado mais fiel quando a imagem em HD é utilizada. Como conclusão, ambos os processos devem ser realizados na borda.

- A partir da linha 8 temos um alto paralelismo dos processos em sequência pela divisão da nuvem em diversas folhas com a octree. Não faz sentido paralelizar a nuvem e em meio aos processos das linhas seguintes enviar a nuvem à névoa, pois isso se resumiria a unificar todas as folhas novamente para serem enviadas pela rede, e obrigando uma nova rodada do Algoritmo de Octree Iterativa no nó em névoa.

- A partir da linha 9 temos também 37,25% do valor total em PU. Dessa forma, não só o processador em *fog* é mais rápido em termos de unidade de tempo real, como também possuirá tempo relativo para compensar o *delay* existente na transferência dos dados pela rede, bem como processar dados enquanto a próxima cena de aquisição é posicionada. O nó em *edge* precisaria se preocupar, então, em aquisitar os dados por 50,13% do tempo total, reduzindo o processamento a somente 10,85% do total apresentado.

Frente a todos os pontos, optou-se pela divisão onde basicamente os dados dos diversos sensores serão capturados em *edge*, sendo submetidos à filtragem básica, salvamento da imagem HD e coloração da nuvem de pontos. A partir daí a nuvem de pontos, imagem em baixa resolução e odometria são descarregados para o nó em névoa, onde todo o processamento paralelo mais robusto ocorre, bem como salvamento das nuvens de pontos finais. A Figura 7 ilustra essa divisão e escalonamento dos níveis do processo.

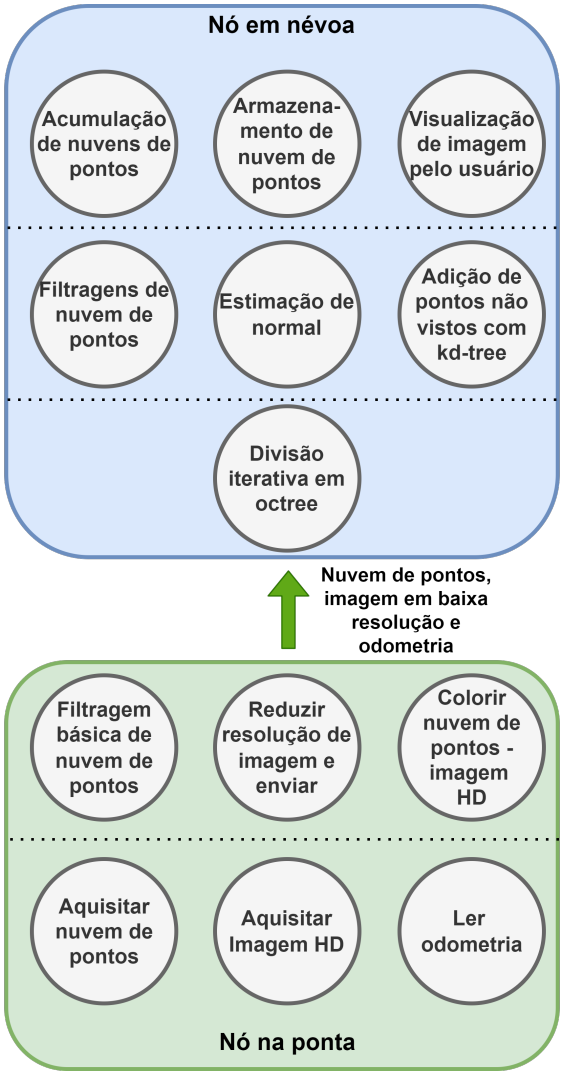


Figura 7: Organização lógica dos processos, de formas escalonada, em ambos os nós.

3.3 ROBÔ DESENVOLVIDO

O robô foi desenvolvido com o intuito de escanear ambientes e objetos em 3D, obtendo uma nuvem de pontos coloridos, e hoje já possui duas versões. A escolha dos componentes ocorreu em paralelo à organização da arquitetura *edge-fog* proposta. Scanners com este intuito normalmente funcionam com uma arquitetura baseada em *edge*, embarcando tudo no próprio equipamento, sendo o operador responsável pelo controle e monitoramento, recebendo *feedbacks*. Todo o resultado deveria ser então pós processado, o que difere do robô proposto, o qual graças à arquitetura fornece o resultado final em quase tempo real a partir do início da operação.

A primeira versão do robô possui um LiDAR 3D da marca Livox, modelo MID-40, com uma tecnologia em estado sólido que reduz drasticamente seu preço a uma precisão

equiparada a outros sensores no mercado [43]. Possui também câmera HD modelo C925s, servos Dynamixel AX-18A, uma IMU modelo MPU-6050, roteador WiFi Gbyte e alimentação com bateria 12V 8400mAh. O processador embarcado é a NVidia Jetson Nano, rodando Ubuntu 18.04 e *Robot Operating System* (ROS) Melodic como sistema operacional e *middleware* de sincronização, e sua justificativa constará na seção de resultados. A Figura 8a ilustra o desenho CAD do robô, enquanto em 8b está uma imagem do mesmo em teste de campo.

(a) (b)

Figura 8: Robô desenvolvido (a) Desenho em CAD; (b) Aplicação da primeira versão durante teste em subestação.

A segunda versão do robô difere em aspectos da carcaça, principalmente para embarcar a nova versão do LiDAR 3D com modelo Horizon, que possui um campo de visão maior e IMU integrada, com melhor qualidade de captura. A Figura 9 ilustra esse robô em atuação em uma Usina Hidrelétrica.

O esquema da Figura 10 ilustra as ligações internas do robô, tanto a parte de potência quanto sensorial, de forma geral.

Devido ao formato de construção, o robô possui três regiões maiores: a parte inferior, chamada base; a parte de ligação, com hastes laterais, entre a parte inferior e superior, chamada pescoço; e a própria parte superior, denominada cabeça. O mesmo se movimenta circularmente em dois graus de liberdade, em torno do eixo vertical do tripé (*pan*) e do eixo da cabeça (*tilt*). Assim, pode percorrer uma vista 360° em diversas vistas, definidas como poses a partir de seus ângulos de *pan* e *tilt*. A combinação de *pan* e *tilt* gera uma orientação, denominada "pose" do robô.

A metodologia de funcionamento do robô consiste em uma missão percorrendo um caminho pré-definido de poses, no qual cada pose gera um *waypoint* deste caminho. O robô faz primeiro vários movimentos em *tilt*, para depois se alterar em *pan*. Daqui em diante, todo movimento em um só grau de liberdade será definido como viagem.

Figura 9: Segunda versão do robô em uso em uma Usina Hidrelétrica.

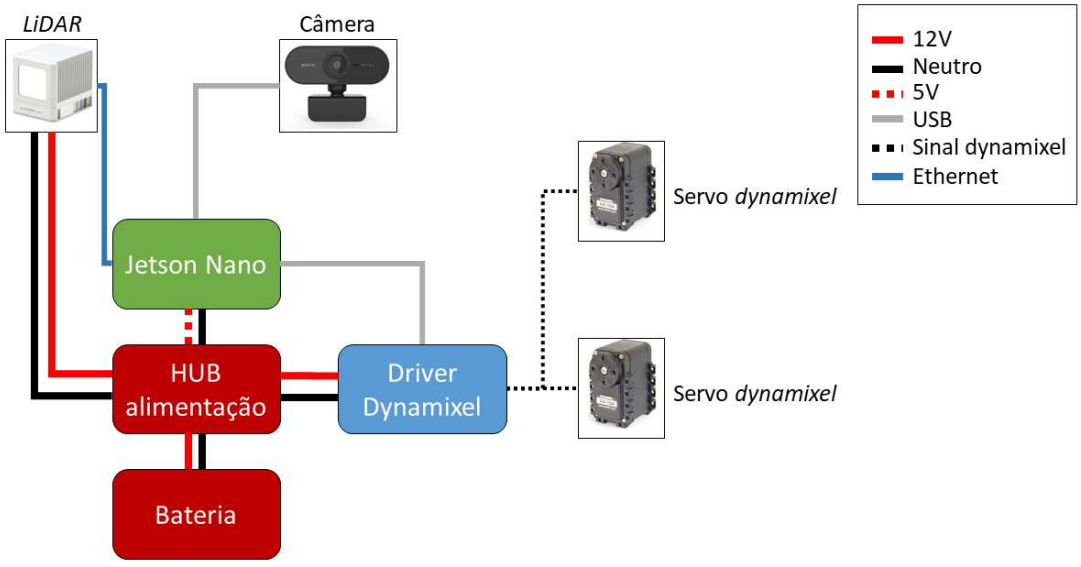


Figura 10: Ligações internas do robô, com sensores, cabos de dados, *drivers* e alimentação.

Da mesma forma, se um grau de liberdade está fixo, temos um ponto de vista (PV). Portanto, se em *pan*, temos um PVP, e se em *tilt*, um PVT.

O robô possui um *frame* local $\mathbb{F}_{lo}$, com a origem coincidente a $\mathbb{F}_{li}$, uma vez que a

origem do segundo se encontra sobre ambos os eixos de rotação. Cada *waypoint* realiza uma captura de nuvem de pontos colorida, que deve ser transformada para $\mathbb{F}_{lo}$. A princípio, dentro de uma viagem em *tilt*, os ângulos de *tilt*, θ e *roll*, ϕ de cada *waypoint* são obtidos por meio da IMU e servos, e usados na matriz de rotação ${}^{PVP}\mathbf{R}_{li} \in \mathbb{SO}_3$ da Equação 3.12a. O leitor pode se referir novamente à Figura 4 para representação dos ângulos sobre os eixos de rotação.

Supondo a nuvem C_{li} no *frame* do LiDAR, cada ponto $\mathbf{P}_{li} \in C_{li}$ é rotacionado para formar a nuvem C_{PVP_i}, como na Equação 3.12b. A simples união das nuvens em um mesmo PVP resulta na nuvem C_{PVP}.

$$
{}^{PVP}\mathbf{R}_{li} = \begin{bmatrix} \cos\phi & -\cos\theta \cdot \operatorname{sen}\phi & \operatorname{sen}\theta \cdot \operatorname{sen}\phi \\ \operatorname{sen}\phi & \cos\phi \cdot \cos\theta & -\cos\phi \cdot \operatorname{sen}\theta \\ 0 & \operatorname{sen}\theta & \cos\theta \end{bmatrix} \tag{3.12a}
$$

$$
C_{PVP_i} = \bigcup \left({}^{PVP}\mathbf{R}_{li} \cdot P_{li_k} \right), \forall P_{li_k} \in C_{li} \tag{3.12b}
$$

Ao final de um PVP, o ângulo de *pan* ψ é aplicado à matriz de rotação ${}^{lo}\mathbf{R}_{PVP} \in \mathbb{SO}_3$ (Equação 3.13a) para transformar a nuvem para o *frame* local. De forma semelhante à Equação 3.12b, obtém-se a nuvem de pontos $C_{PVP_{lo}}$, rotacionando os pontos $\mathbf{P}_{PVP} \in C_{PVP}$, como definido na Equação 3.13b.

$$
{}^{lo}\mathbf{R}_{PVP} = \begin{bmatrix} \cos\psi & 0 & \operatorname{sen}\psi \\ 0 & 1 & 0 \\ -\operatorname{sen}\psi & 0 & \cos\psi \end{bmatrix} \tag{3.13a}
$$

$$
C_{PVP_{lo}} = \bigcup \left({}^{lo}\mathbf{R}_{PVP} \cdot P_{PVP_k} \right), \forall P_{PVP_k} \in C_{PVP} \tag{3.13b}
$$

As nuvens C_{wp}, C_{PVP} e $C_{PVP_{lo}}$ são submetidas aos processos de filtragem e processamento descritos ao longo deste Capítulo. Como fim, a união de todas as nuvens $C_{PVP_{lo}}$ resulta na nuvem registrada final, ou acumulada, C_{lo}, em $\mathbb{F}_{lo}$.

3.4 FLUXOGRAMAS DE FUNCIONAMENTO

A divisão escalonada da Figura 7 e a aplicação das técnicas algorítmicas e matemáticas deste Capítulo serão expostas em detalhes em fluxogramas para os nós das camadas de *edge* e *fog*.

3.4.1 FLUXO DE PROCESSAMENTO NA BORDA

O fluxograma da Figura 11 ilustra o processo que ocorre no nó em *edge*. Ao iniciar o sistema e a missão de escaneamento, sempre checa-se em qual *waypoint* se encontra o robô. Supondo n_t PVTs e n_p PVPs, temos um total de $n_p \cdot n_t$ *waypoints* na missão. Ao chegar em um *waypoint*, o robô aquisita a nuvem de pontos, imagem e leitura de odometria. Após isso, realiza filtragens básicas, salva a imagem e reduz a resolução da mesma para envio. Após processar a nuvem de pontos colorida, os dados são enviados para a camada em névoa. Quando alcançamos o *waypoint* final pré-calculado, já tendo varrido os 360 graus em torno do robô, finaliza-se o algoritmo.

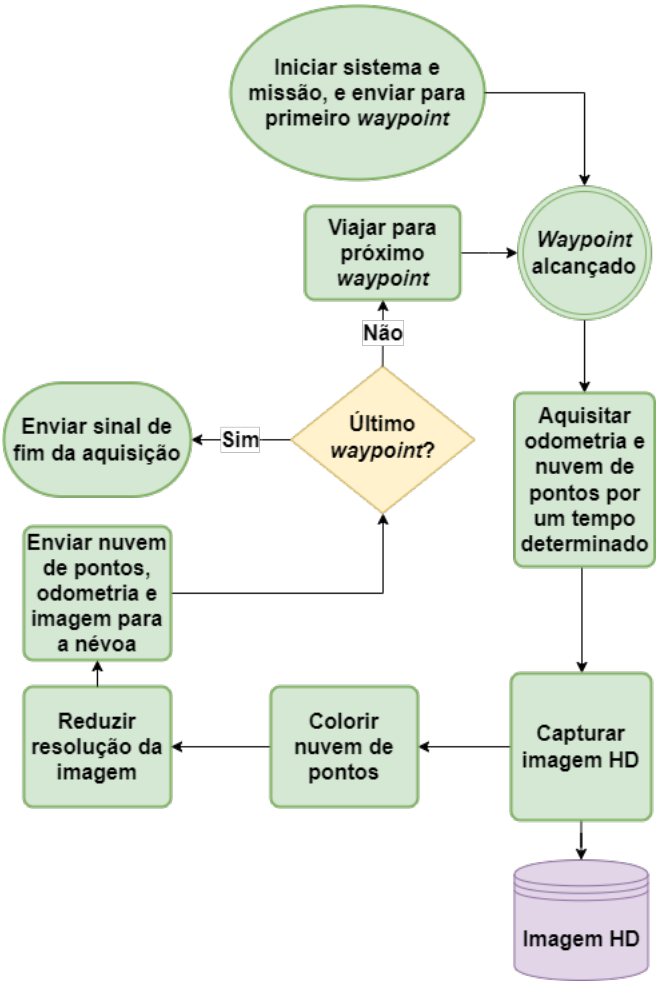


Figura 11: Fluxograma do processo na borda.

3.4.2 FLUXO DE PROCESSAMENTO EM NÉVOA

O fluxograma da Figura 12 ilustra o processo que ocorre no nó em *fog*. O ciclo se inicia nos nós em verde, com os dados vindos da borda para imagem, nuvem de

pontos e odometria. A imagem segue diretamente para a interface com o usuário. A nuvem de pontos é submetida ao Algoritmo de Octree Iterativa, e em sequência todos os filtros mencionados são aplicados: *ray casting*, covariância e SOR. A nuvem sofre as transformações por meio da odometria para ir do *frame* $\mathbb{F}_{li}$ para $\mathbb{F}_{lo}$. Neste ponto, a nuvem atual pode ser registrada à nuvem das aquisições anteriores, criando a nuvem acumulada por meio do registro de pontos não vistos com kd-tree. A nuvem acumulada também pode ser vista na interface com o usuário. Caso seja alcançado o último *waypoint*, a nuvem e as poses de toda a missão são salvas no próprio nó em névoa.

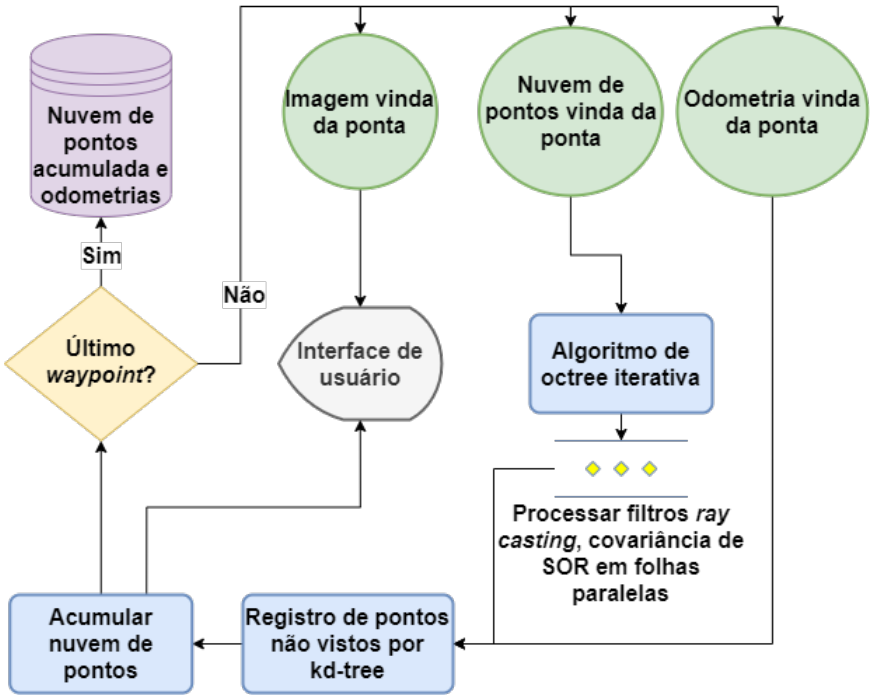


Figura 12: Fluxograma do processo em névoa.

4 RESULTADOS EXPERIMENTAIS E DISCUSSÃO

Este capítulo apresenta os resultados, focando nas principais contribuições deste trabalho em termos de inovação da aplicação:

- Escolha dos processadores em *edge* e *fog*, bem como seus valores em termos reais.
- Demonstração da melhoria de resultados utilizando o Algoritmo de Octree Iterativa, em comparação com a octree original, e o paralelismo proposto;
- Comparação da aplicação quando embarcada totalmente na borda, como visto em outros trabalhos citados, e a arquitetura *edge-fog* proposta, ressaltando sua possibilidade de aplicação em tempo-real;
- A escalabilidade da solução, a partir de análise de até 5 robôs funcionando em paralelo.

Os novos dados utilizados para obter os resultados foram obtidos da inspeção de um cenário real de aplicação, na Usina Hidrelétrica de Peixe Angical, local onde foi capturada a Figura 9.

4.1 DESCRIÇÃO DOS PROCESSADORES UTILIZADOS

A escolha para o processador em *edge* foi da placa NVidia Jetson Nano. A mesma apresenta processador Quad-core ARM A57, 4Gb de memória RAM e armazenamento por cartão SD de até 250Gb, além de GPU integrada. A escolha se baseou na capacidade de memória, que permite o uso do sistema operacional e processamentos locais sem travamento do computador, com um processador de capacidade relevante em comparação com outras placas embarcadas. Por fim, seu custo é inferior ao de processadores mais robustos, como NUCs, e há compatibilidade com os sensores utilizados na solução. Ao testar o processador quanto à PU, foi encontrado um valor de 95 nanosegundos.

O computador concebido como nó de processamento em névoa possui um Intel Quad-core i5-2200U@2.20GHz, 8Gb de memória RAM e 500Gb de armazenamento em SSD. Este computador apresentou um PU de 72 nanosegundos.

Em posse de dados de ambos os processadores, é iniciada a alocação de processos como proposto na Seção 3.2.3. É observado na prática que os núcleos do processador em *edge* atingem 100% de utilização quando iniciado o processo de filtragens mais robustas em paralelo (da Tabela 2, a partir do Algoritmo de Octree Iterativa na linha 8). Isso corrobora a justificativa de ser esse um ponto de divisão entre as arquiteturas, e que a escolha da alocação está correta com a capacidade de processamento de cada entidade.

Em unidade de tempo, os dados deveriam demorar 3,12 segundos para deixar a camada de borda e 0,94 segundos para serem processados na camada de névoa, em média, para cada aquisição. Há portanto tempo de sobra também para possíveis inconsistências na rede ao longo da execução, algo que não ocorreu normalmente nos testes apresentados a seguir.

4.2 ALGORITMO DE OCTREE ITERATIVA

Para teste do Algoritmo de Octree Iterativa, o nó em névoa foi submetido a testes em quatro cenários, como citados abaixo. Em todos eles, cada nuvem de pontos é encaminhada pelo robô como uma única mensagem e decodificada.

- **Cenário 1:** A nuvem de pontos de entrada é processada da forma como foi enviada pelo robô, sem nenhuma divisão ou processamento paralelo de folhas.
- **Cenário 2:** A mesma nuvem é submetida à divisão por octree tradicional, ajustando a profundidade supondo a nuvem de entrada homogênea, até que a densidade chegue a um nível satisfatório.
- **Cenário 3:** A nuvem de pontos é submetida ao algoritmo proposto, até que todas as folhas possuam um limite máximo de pontos, usando a divisão geométrica.
- **Cenário 4:** A nuvem de pontos é submetida ao algoritmo proposto, até que todas as folhas possuam um limite máximo de pontos, usando a divisão por cálculo de centróide.

Em todos os cenários, os processos avaliados foram os filtros que devem ser performados em névoa: *ray casting*, covariância e SOR, bem como a estimação de normais. A

Tabela 3 indica a média dos tempos, em PU, para cada processo nos cenários descritos.

Tabela 3: Tempo para cada processo no nó em névoa, em PU($x10^5$), para os cenários propostos.

Processo	**Cenário 1**	**Cenário 2**	**Cenário 3**	**Cenário 4**
Divisão em octree	-	6,31	9,72	7,52
***Ray casting* e covariância**	117,90	74,12	59,75	49,22
SOR	23,82	12,91	9,51	9,02
Estimar normais	51,20	24,06	19,30	14,43
Total	**196.92**	**119.40**	**98.28**	**80,19**

Como pode ser visto pela coluna central, o tempo total para o algoritmo de octree tradicional do Cenário 2 reduziu em aproximadamente 40% o tempo total inicial, porém dois processos tiveram melhora em 50%, valor próximo ao apresentado pelo trabalho de [25]. Já para o Cenário 3, essa melhora foi ainda maior, reduzindo em 50% o tempo total, e até 62,3% no caso da estimação de normais. Houve benefício em todos os processos, ao custo de um aumento de tempo para o algoritmo de divisão em octree, que é irrelevante quando comparado ao tempo tomado pelos mesmos.

Para reforçar o benefício obtido no cenário 4, primeiramente observamos a Tabela 3. Vemos a princípio uma redução no tempo de duração do algoritmo, provavelmente devido à expectativa de um menor número de iterações. Logo em sequência vemos uma melhora no tempo de processamento de todos os algoritmos, também teoricamente devido ao melhor balanceamento das nuvens de pontos resultantes, o que evita o gargalo que alguma *thread* com quantidade maior de pontos possa enfrentar e atrasar o fluxo de processamento.

Tendo observado isso, foi estudado a fundo a utilização dessa nova técnica de separação em octree frente ao *stream* de dados obtido diretamente do LiDAR, independentemente da nossa arquitetura proposta. As seções das nuvens de pontos de entrada também foram controladas, contendo entre 1000 e 2000 pontos, para poder observar claramente o comportamento dos algoritmos. O tamanho máximo de cada folha foi definido em 100 pontos.

A Figura 13 ilustra o tomado pelo Algoritmo de Octree Iterativa original, e com a adição do cálculo de centroides. O gráfico corrobora com o outro resultado apresentado na Figura 14, contendo o número de iterações para cada algoritmo, e dá suporte ao que era esperado em teoria: o cálculo de centroides necessita de menos iterações, e portanto é obtido mais rapidamente.

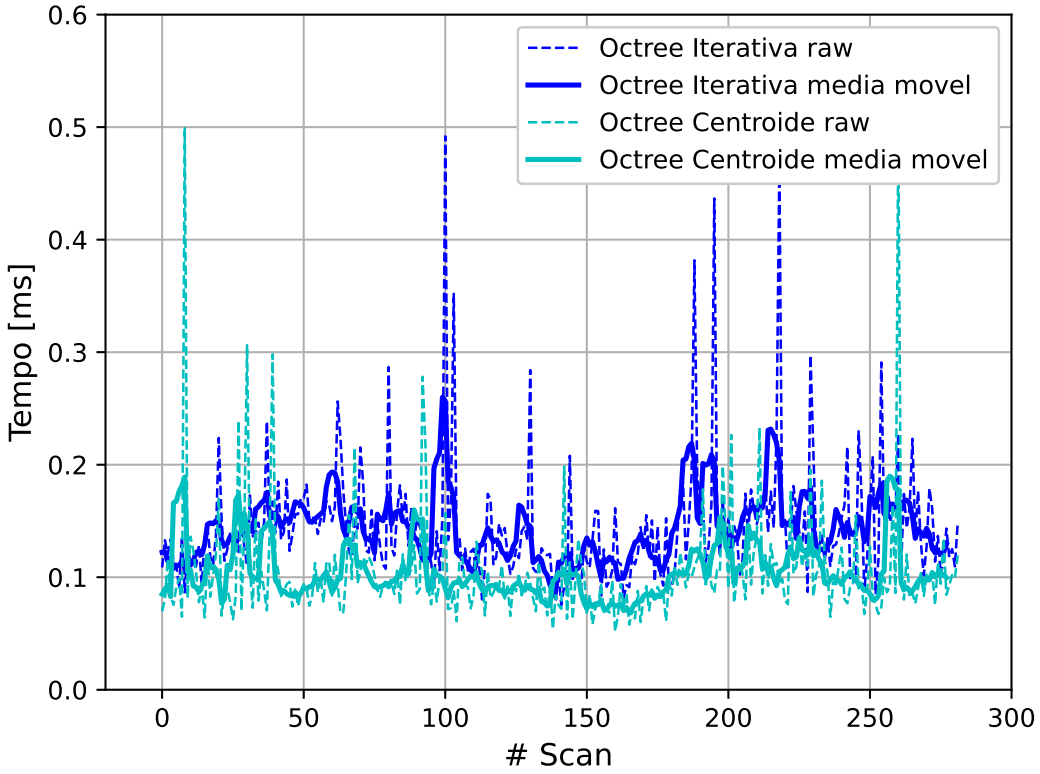


Figura 13: Tempos para divisão das nuvens de pontos de entrada pelas duas versões do algoritmo de octree, em milissegundos.

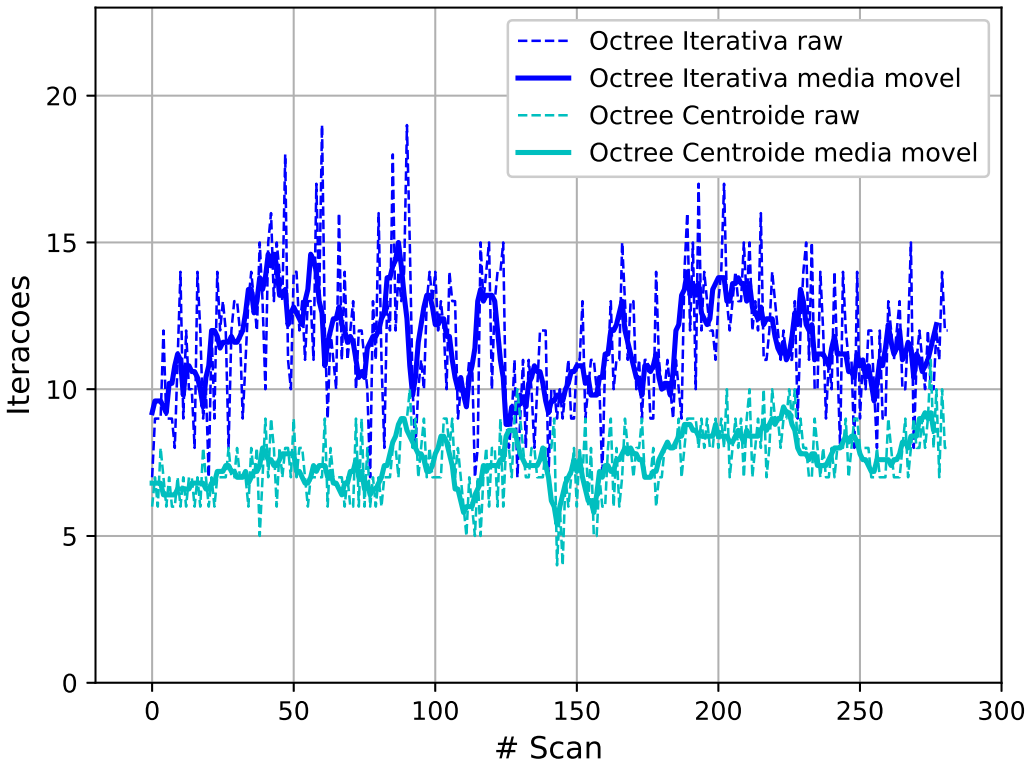


Figura 14: Número de iterações para ambas as versões do Algoritmo de Octree Iterativa.

Por fim, a Figura 15 mostra o tempo de execução para o cenário sem qualquer processo paralelo, com a Octree Iterativa, e a versão com cálculo de centroide, frente aos processos propostos para o nó em névoa e citados na Tabela 3. Pode-se ver claramente o benefício no tempo de processamento com o cálculo de centroides.

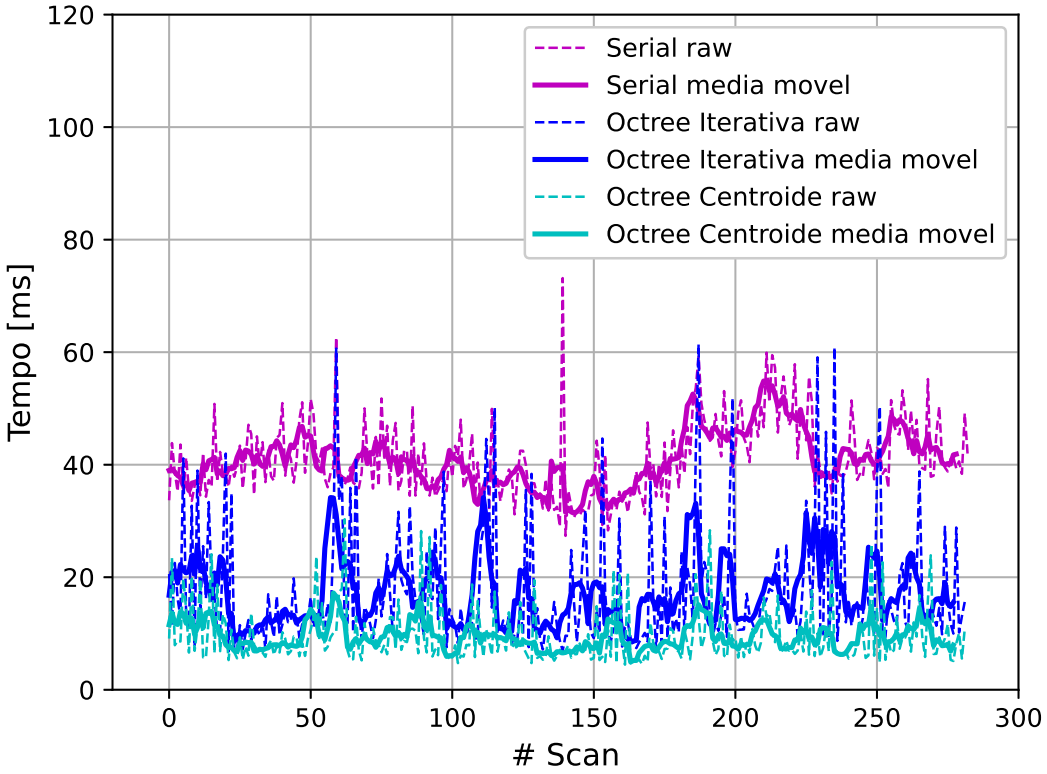


Figura 15: Tempos para execução de todos os algoritmos propostos na Tabela 3, de forma serial ou com as versões do algoritmo de octree, em milisegundos.

4.3 COMPARAÇÃO ENTRE ARQUITETURAS

Esta seção introduz o tópico de maior contribuição deste trabalho, uma vez que resultados em aplicações práticas ainda são reduzidos na literatura no que diz respeito a arquitetura *edge-fog* e suas derivações, como mostrado no Capítulo 2. Trabalhos como [12, 46] propõem a base do comportamento esperado baseado em simulações de entes na rede nas diversas camadas em *edge*, *fog* e *cloud*. Outros trabalhos apresentam protótipos para basear suas propostas teóricas [34, 31, 32, 47]. Ainda assim, não se vê uma análise de dados tão completa como mostrada a seguir, bem como estudo de escalabilidade, que será apresentado na próxima Subseção 4.4.

Buscando uma solução em tempo real e escalável, duas eram as possibilidades: basear a solução em processamento na borda, sendo todos os processos executados pelo robô, e o usuário poderia monitorar através de uma interface o trabalho de cada um, ou arquitetá-los em um formato *edge-fog*, que é a proposta deste trabalho. Ao obter suporte da computação em névoa, a borda pode descarregar dados e processamento para

um nó centralizado mais capacitado para a funções, facilitando também o sincronismo entre as aquisições.

O primeiro dado a ser comparado diz respeito à latência obtida em ambas as arquiteturas. Baseado em [48], a latência é assumida como o tempo de espera entre a coleta de dados e a visualização dos mesmos pelo usuário. O tempo médio para transferência de dados de nuvem e imagem em cada *waypoint* foi de 1,36 e 1,82 segundos para as arquiteturas *edge-fog* e *edge-based*, respectivamente. Isso se deve ao fato de, no segundo caso, a nuvem de pontos ser enviada após sofrer todos os processos de filtragem propostos.

A segunda análise referente à rede diz respeito ao *throughput*. O mesmo foi medido durante a aquisição para ambas as propostas de arquitetura, para todos os conjuntos de teste. O gráfico da Figura 16 ilustra um minuto do comportamento da taxa de transferência de rede para ambas as arquiteturas, e será debatido a seguir.

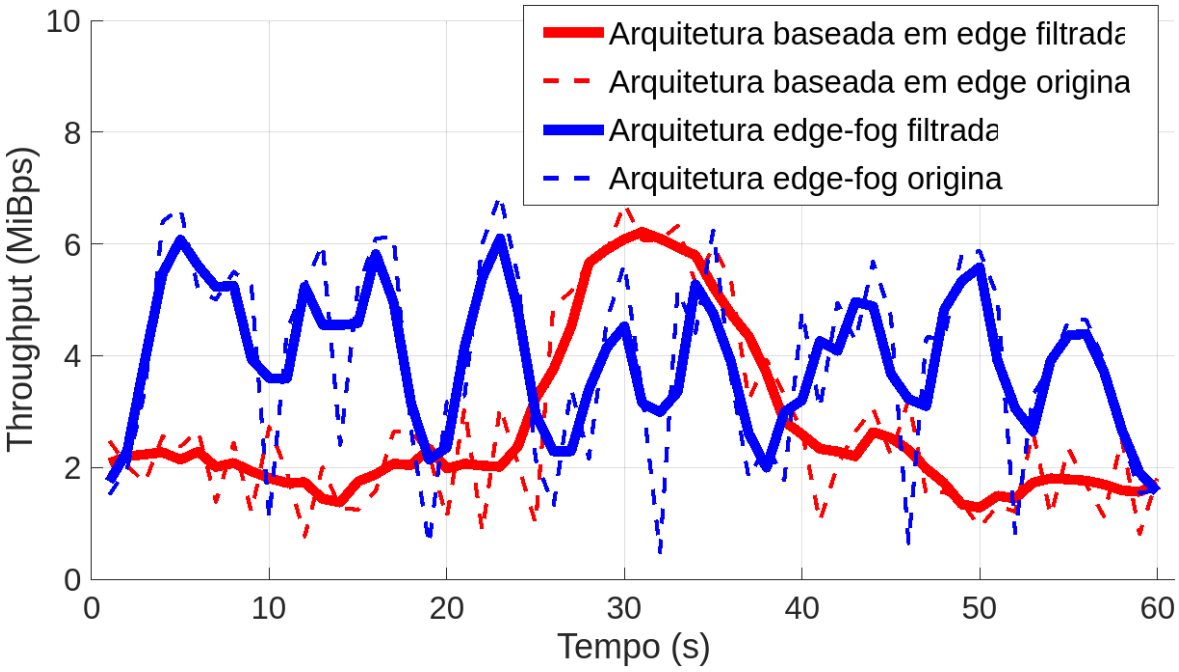


Figura 16: Throughput para ambas as arquiteturas durante um minuto da aquisição. Linhas tracejadas representam o dado real, enquanto linhas sólidas mostram a média móvel.

Na arquitetura baseada em *edge* temos um processamento completo da nuvem de pontos antes do envio, o que resulta em um uso da rede esparso e ineficaz, forçando o pico de uso no momento da transferência, como visto nos gráficos em vermelho da Figura 16. Um dos grandes benefícios da arquitetura proposta consiste no uso

balanceado da rede, transmitindo constantemente dados filtrados de cada aquisição, e liberando ambos os nós para a sequência do trabalho após o envio dos mesmos.

Os próximos valores analisados dizem respeito ao gasto energético e desempenho de CPU, bem como uso de memória RAM e armazenamento. O uso de CPU leva em consideração a média de trabalho de todos os núcleos em cada nó, *edge* e *fog*. O consumo energético e desempenho de CPU foram medidos com os programas nativos do Ubuntu *powerstat* e *mpstat*, respectivamente. Nenhum trabalho de GPU é requerido em ambas as máquinas. Os resultados estão postos na Tabela 4.

Tabela 4: Atividade de CPU, consumo energético, pico de uso de memória RAM e armazenamento para os dados de teste em ambas as arquiteturas.

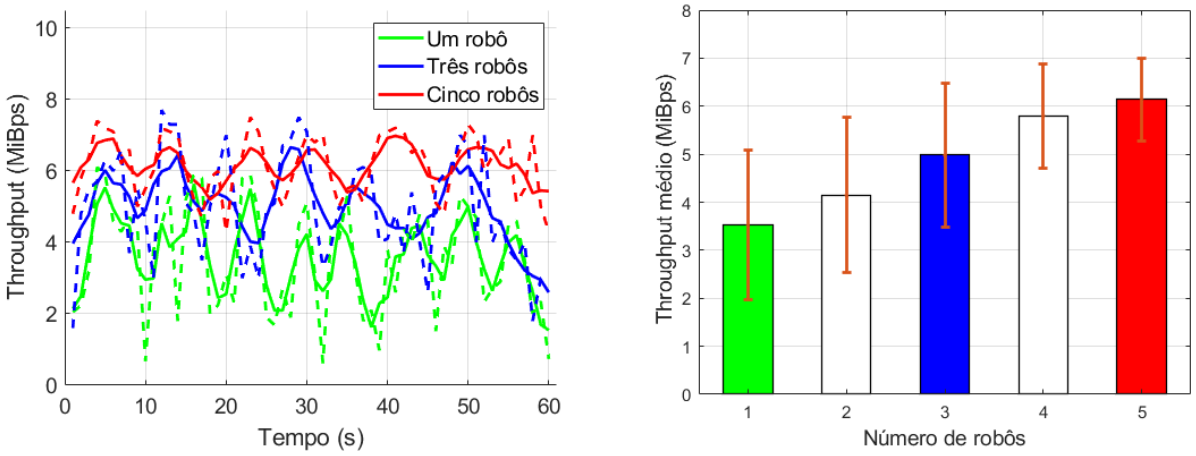
	Arquitetura baseada em *edge*		**Arquitetura *edge-fog***	
	Edge	Interface usuário	*Edge*	*Fog*
Atividade de CPU (%)	91,4	-	67,2	32,9
Potência media (W)	6.92	14,0	6,13	16,0
RAM (%)	88,3	-	34,8	7,9
Armazenamento (MB)	563	-	243	320

Como principais resultados, a Tabela 4 demonstra um maior balanceamento quanto a uso de CPU, com uma redução de 24,1% na borda. A necessidade de armazenamento foi reduzida em um terço na borda, algo que permite ao usuário performar mais escaneamentos em campo, a detrimento de um aumento de armazenamento considerado irrelevante no nó em névoa.

4.4 ESCALABILIDADE DA SOLUÇÃO

Um dos pontos mais relevantes em termos de benefício do uso de *Fog computing* é a escalabilidade da solução, frente a uma arquitetura baseada na borda. Neste trabalho em particular, a presença de um nó, ou possivelmente mais de um, na camada em névoa, também possibilita o sincronismo e disposição em tempo real dos resultados obtidos. Dessa forma, até cinco aquisições foram performadas em paralelo, e analisadas a seguir.

A primeira análise diz respeito ao impacto na rede pelo uso de robôs em paralelo na borda. Para ilustrar o aumento de uso da mesma, o gráfico da Figura 17a traz a curva de *throughput* para 1, 3 e 5 robôs. Os valores de 2 e 4 foram omitidos para simplificar a visualização, mas é possível ver a tendência de comportamento. Os valores de média e desvio padrão para as curvas estão ilustrados da Figura 17b. A tendência de uso da rede claramente cresce, buscando a saturação em aproximadamente $8MiBps$.



(a) (b)

Figura 17: Evolução do *throughput* para um cenário de um a cinco robôs, onde linhas tracejadas e sólidas representam os dados reais e as médias móveis, respectivamente: (a) Curvas para um minuto de aquisição (b) Média mais desvio padrão.

A próxima análise que se segue diz respeito a uso de CPU e pico de memória RAM por parte do nó em *fog*, para 1 a 5 robôs funcionando em paralelo. A Tabela 5 mostra os valores obtidos.

Tabela 5: Atividade de CPU e pico de memória RAM no nó em névoa para até 5 robôs funcionando em paralelo.

	Número de robôs				
	1	**2**	**3**	**4**	**5**
Atividade de CPU (%)	30,2	33,6	42,9	53,3	63,7
RAM (%)	5,9	11,0	16,2	19,9	27,1

Nota-se o comportamento crescente de todos os dados à medida que aumenta o número de robôs observados. Porém, ambos os dados estão aquém do valor máximo disposto pela máquina, ao passo que o processo encontra gargalos de 100% de uso do processador quando executado baseado em *edge*. Seguindo a tendência de crescimento do uso de CPU, cada nó em névoa encontraria um gargalo quando recebendo dados de 9 nós na camada de borda.

Tabela 6: Tempo total para o escaneamento completo em cada cenário avaliado.

Arquitetura e cenário	**Tempo total (s)**
Baseada em *edge* (cada robô)	585,3
Edge-fog com um robô	498,1
Edge-fog com dois robôs	505,2
Edge-fog com três robôs	510.5
Edge-fog com quatro robôs	560,1
Edge-fog com cinco robôs	576.1

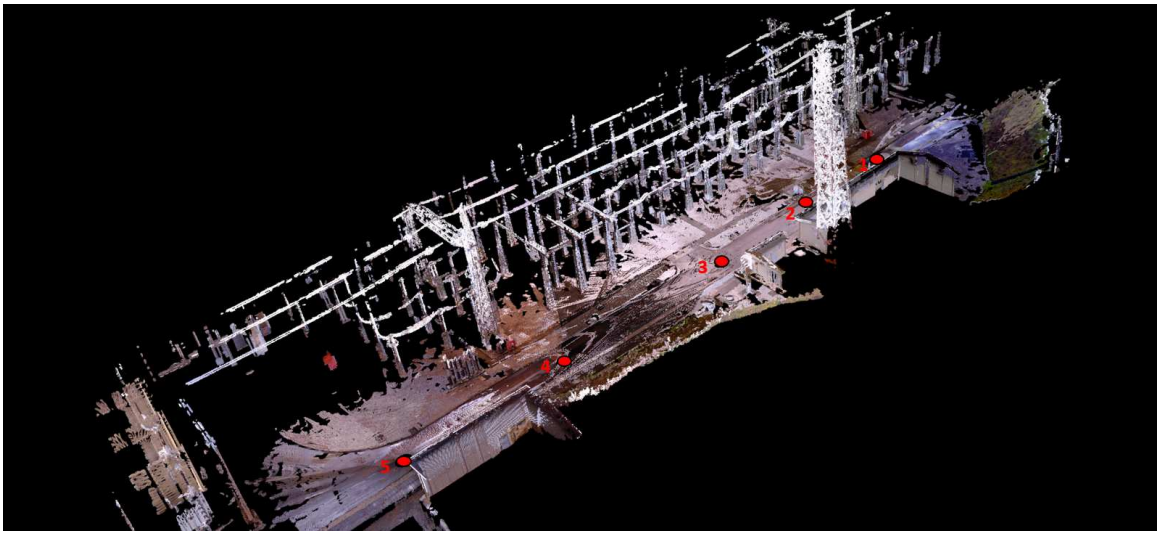
Da Tabela 6 pode-se observar que a arquitetura proposta foi capaz de realizar 5 escaneamentos completos com tempo inferior ao requerido por somente um, no caso da arquitetura baseada no processamento na borda. Mesmo com o nó em névoa não enfrentando gargalos de processamento, como já mostrado na Tabela 5, o crescimento do tempo total ocorreu pelo congestionamento da rede, ilustrado na Figura 17, e que se torna o gargalo com a tecnologia e *hardware* utilizados. Além disso, é reforçado novamente que todos os dados de aquisição já se encontravam sincronizados em um único nó em *fog*, sendo possível a fusão dos mesmos em tempo real.

A mostra qualitativa dos resultados obtidos de reconstrução na subestação de Santos Dumont, já apresentados no momento de qualificação desta tese de doutorado, estão ilustrados as Figuras 18a-c, a partir de 5 pontos de aquisição na subestação, marcados em vermelho na Figura 18a.

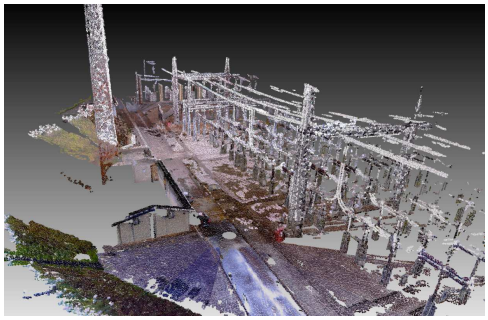
Para os novos resultados apresentados obtidos na hidrelétrica de Peixe Angical, as Figuras 19a-b ilustram a reconstrução da sala de máquinas e da sala de turbina, tanto a imagem obtida, quanto a nuvem de pontos após o processamento em *fog*.

4.5 ESTUDO DE INSERÇÃO DA SOLUÇÃO NO CENÁRIO REAL DE INSPEÇÃO DA HIDRELÉTRICA EM UM CONTEXTO DE REDE DE COMUNICAÇÃO 5G

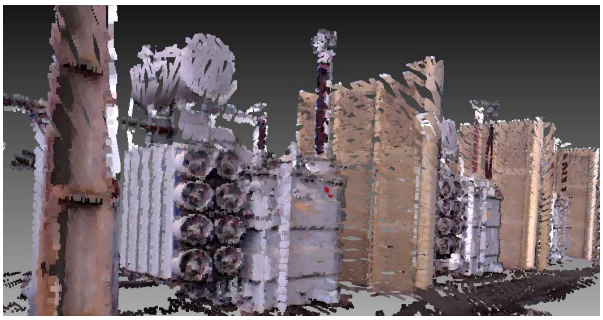
Nosso hardware proposto atual usa uma rede Wi-Fi local para comunicação entre os nós. Os resultados confirmaram que esta tecnologia é limitada em termos de largura de banda e taxa de transferência. Os recursos de rede tornam-se o gargalo em termos de escalabilidade à medida que o número de robôs paralelos aumenta. Conforme apresentado em [37], a rede 4G poderia ser uma solução com maior alcance de sinal, mas ainda carece principalmente de largura de banda, *downlink*, e recursos de *uplink*



(a)



(b)



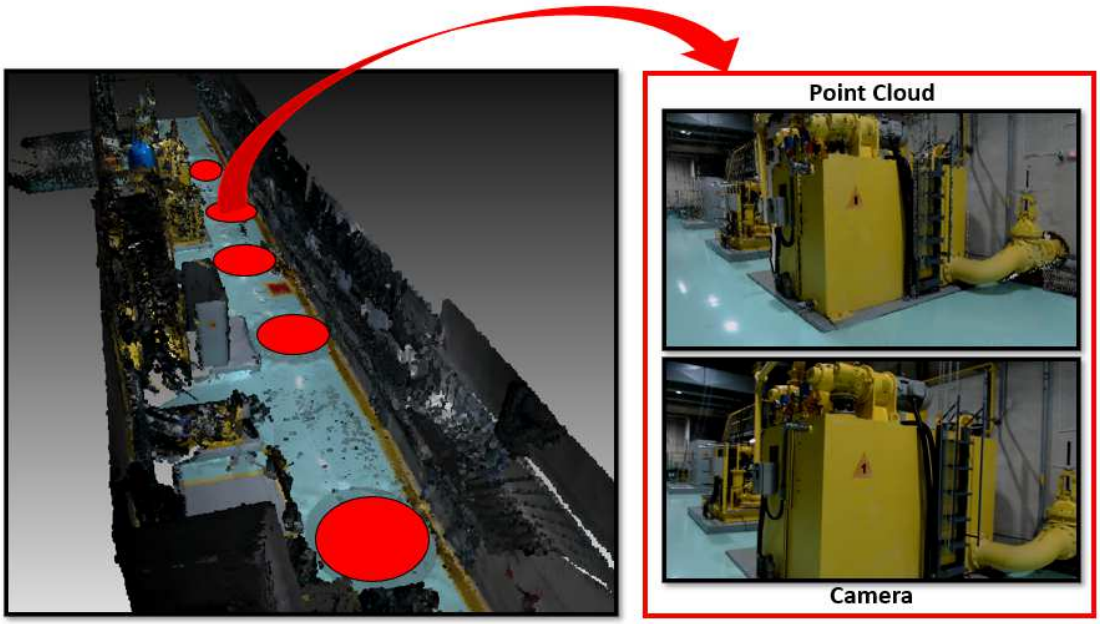
(c)

Figura 18: Estrada lateral da subestação em nuvem de pontos. (a) Vista azimutal com posições de aquisição em vermelho; Vista azimutal do segundo quadrante; (c) Banco de transformadores ao fundo.

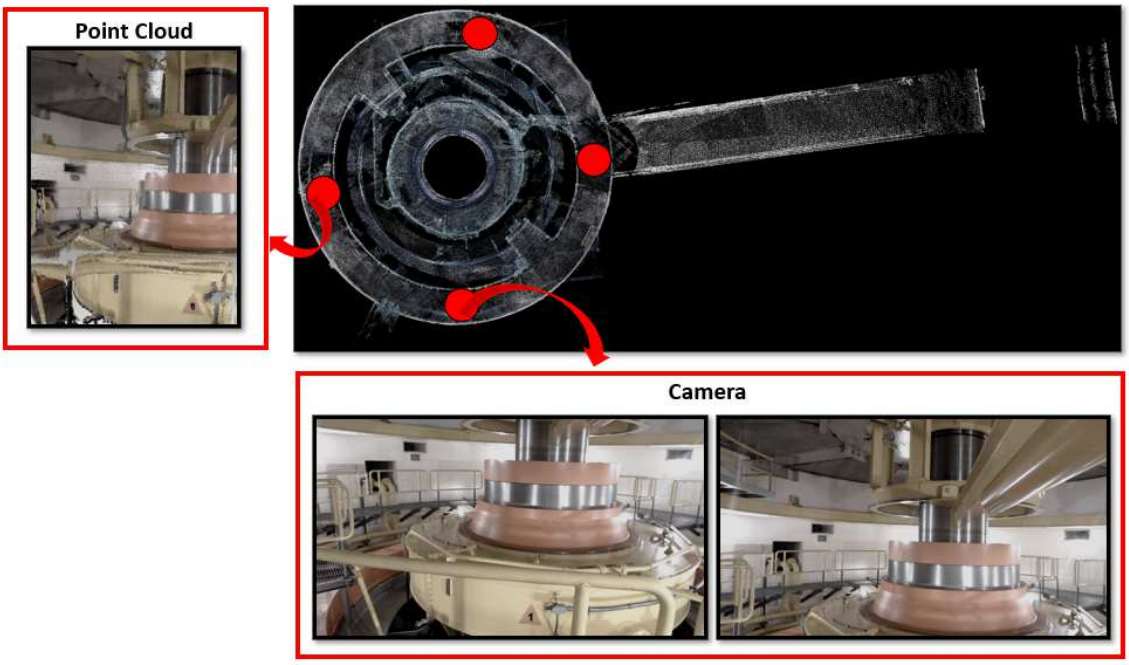
para atender uma usina inteira no contexto atual de controle remoto e monitoramento envolvendo imagem e reconstrução 3D. Assim, o 5G surge como a melhor opção para esta aplicação.

Várias referências apontam para redes reais sendo aplicadas em cenários industriais, descrevendo os recursos de rede, como será mostrado a seguir. Nosso trabalho está interessado principalmente no *downlink* e capacidades de *uplink* de redes 5G privadas, uma vez que evitará o atual problema de saturação da rede devido à baixa largura de banda.

O trabalho de [49] avalia o comportamento de uma rede de estação base 5G comercial quando comunicando-se com dispositivos de bordo em drones voando em diferentes alturas. Eles testemunharam valores de até 742 Mbps e 46 Mbps para taxas de *downlink* e *uplink*, respectivamente. Baseado neste esforço, [37] apresenta uma discussão e metodologia sobre alocação de processos em um problema de navegação visual-inercial de



(a)



(b)

Figura 19: Usina hidrelétrica de Peixe Angical (a) Sala de máquinas. (b) Sala da turbina.

drone baseado em características de rede, entre outros. Ele simula taxas de *downlink* e *uplink* variando de 320–6400 Mbps e 40–800 Mbps, respectivamente. Finalmente, num ambiente mais semelhante ao deste trabalho, [50] apresenta a aplicação e testes de uma estação base 5G comercial em uma subestação no Brasil, visando realizar uma prova de conceito para aplicativos de automação de redes e *digital twins*. Para um *laptop*

convencional, os valores médios das taxas de *downlink* e *uplink* foram registrados como 958 Mbps e 83 Mbps, respectivamente.

Com base nesses valores coletados de dados reais, as seguintes previsões são assumidas para aplicação de rede privada 5G no cenário de monitoramento remoto de usina:

- A partir dos dados coletados e mostrados na Figura 17b, como uma abordagem conservadora, espera-se que cada robô exija 5,67 MiBps de *throughput*, considerando a média mais um desvio padrão, que é equivalente a 47,56 Mbps.
- Cada nó de névoa ficará saturado com o processamento de dados de 9 nós de borda devido à restrição de utilização da CPU.
- 9 nós na camada de *edge* paralelos exigirão uma taxa de transferência de $9x47{,}56Mbps = 428{,}04Mbps$ em cada conexão a um nó em *fog*.
- O valor de 83 Mbps para taxa de *uplink* registrado em [49] já é suficiente para que cada nó em *edge* envie dados de acordo com nosso primeiro requisito experimental, então qualquer valor maior só ajudará na robustez da rede e diminuição da latência.

O resultado para o número de nós de névoa e nós de borda que podem funcionar simultaneamente é avaliado de acordo com a capacidade de *downlink* da rede, com o resultado demonstrado na Figura 20. Os resultados são plotados para valores de *downlink* variando de 700–6400 Mbps. A forma como estimamos é simplesmente adicionar um nó na borda uma vez que a largura de banda necessária está disponível, e um nó de névoa quando o número de nós na borda passar a requerer mais um para desafogar o processamento. Consideramos uma média de nove robôs por sala (dados estimados localmente para abranger o principais pontos de vista, ainda permitindo que o nó de névoa esteja fisicamente mais próximo da camada de borda). O número crescente de robôs permitirá que todas as salas principais da usina sejam monitoradas a partir de uma taxa de *downlink* de aproximadamente 3000 Mbps ou superior (sete salas).

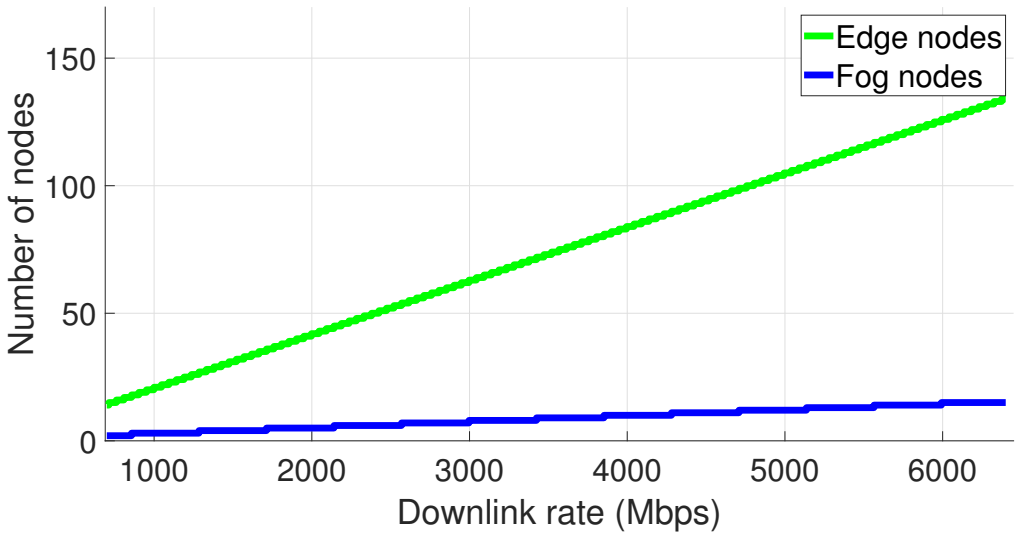


Figura 20: Evolução do número de nós em *edge* e *fog* de acordo com a disponibilidade da taxa de *downlink* da rede privada 5G.

5 CONSIDERAÇÕES FINAIS

5.1 PUBLICAÇÕES RELACIONADAS

Houveram publicações relacionadas ao longo do desenvolvimento deste trabalho, que serviram de base para o que foi apresentado até então, e serão pontuadas abaixo:

Artigos de base teórica:

- BADAT, LEYA ; Vidal, Vinicius ; PIOLI, LAÉRCIO ; MEHAUT, JEAN-FRANCOIS ; HONORIO, LEONARDO ; DANTAS, MARIO ANTONIO RIBEIRO . An IIoT Edge Environment as a Main Support to a 3D Reconstruction Virtualization Application. In: XVIII Workshop em Clouds e Aplicações, 2020, Brasil. Anais do XVIII Workshop em Clouds e Aplicações (WCGA 2020). p. 1.
- VIDAL, VINICIUS F.; HONÓRIO, LEONARDO M. ; DIAS, FELIPE M. ; PINTO, MILENA F. ; CARVALHO, ALEXANDRE L. ; MARCATO, ANDRE L. M. . Sensors Fusion and Multidimensional Point Cloud Analysis for Electrical Power System Inspection. SENSORS, v. 20, p. 4042, 2020.
- SILVA, LUIZ A. Z. DA ; VIDAL, VINICIUS F. ; HONÓRIO, LEONARDO M. ; DANTAS, MÁRIO A. R. ; PINTO, MILENA FARIA ; CAPRETZ, MIRIAM . A Heterogeneous Edge-Fog Environment Supporting Digital Twins for Remote Inspections. SENSORS, v. 20, p. 5296, 2020.

Nestes trabalhos foi abordado o uso de visão computacional a partir de câmeras RGB, RGB-D e térmicas em diversos cenários, desde internos a externos em linhas férreas, para fins de manutenção. Foram propostas soluções otimizadas para detecção de defeitos, bem como falsos defeitos. Foram propostas soluções no âmbito de *Fog Computing* para sanar a necessidade de poder computacional, ausente nas camadas de borda, mostradas de forma satisfatória. Esses trabalhos apresentam, então, base da identificação e solução do problema introduzido por este documento.

Capítulo de livro:

- Decamps, Marceau ; Meháut, Jean-Francois ; Vidal, Vinicius ; HONORIO, LEONARDO ; PIOLI, LAÉRCIO ; Dantas, Mario A. R. . An Implementation Science Effort in a Heterogenous Edge Computing Platform to Support a Case Study of a Virtual Scenario Application. Lecture Notes in Networks and Systems. 1ed.: Springer International Publishing, 2021, v. 158, p. 136-147.

Neste capítulo de livro é possível identificar um caso de estudo para um sistema de visão computacional introduzindo o conceito de componentes heterogêneos em diversas camadas, caracterizando uma arquitetura *edge-fog* em um cenário de sensores IoT.

Por fim, este trabalho deu origem a dois artigos de Qualis A1: o primeiro, publicado na revista *Future Generation Computer Systems*, de fator de impacto JCR 7,187, intitulado "*An Edge-Fog Architecture for Distributed 3D Reconstruction*".

- Vidal, V. F., Honorio, L. M., Pinto, M. F., Dantas, M. A., Aguiar, M. J., & Capretz, M. (2022). An edge-fog architecture for distributed 3D reconstruction. Future Generation Computer Systems.

Já o segundo, publicado na revista *Sensors*, de fator de impacto JCR 2021 3,847 (nos últimos 5 anos prévios a 2021, 4.050), intitulado 'An Edge-Fog Architecture for Distributed 3D Reconstruction and Remote Monitoring of a Power Plant Site in the Context of 5G.'.

- Vidal, V., Honório, L., Pinto, M., Dantas, M., Aguiar, M., & Capretz, M. (2022). An Edge-Fog Architecture for Distributed 3D Reconstruction and Remote Monitoring of a Power Plant Site in the Context of 5G. Sensors, 22(12), 4494.

Mais detalhes sobre os trabalhos podem ser vistos no Anexo A.

5.2 CONCLUSÃO

Este trabalho demonstrou uma solução para reconstrução 3D colorida de ambientes e objetos, em especial no ambiente de uma subestação de energia. Por necessidades computacionais, foi proposta uma arquitetura em modelo *edge-fog* para descarregar processadores nas bordas, e distribuir as cargas computacionais e armazenamento entre

duas camadas: a já mencionada borda, e o processamento em névoa. Para isso, a camada em névoa conta com um computador mais potente como nó de suporte, e por estar na rede local garante aplicação em tempo real e baixa latência, em um cenário onde normalmente o processamento é lento por ser processado e armazenado nas bordas, ou é *offline*, inviabilizando o tempo real.

A solução envolveu a construção de um robô dedicado à tarefa na borda, composto por sensores (principalmente LiDAR 3D e câmera) e processador embarcado, em duas versões. Este recebe uma missão de escaneamento, onde obtêm dados, realiza processos menos custosos, e descarrega dados e processos para a névoa, onde o restante é processado e armazenado. Toda a metodologia de funcionamento em ambas as camadas, bem como distribuição das tarefas entre as mesmas, foi apresentada ao longo do trabalho.

Em termos de reconstrução 3D e nuvem de pontos, técnicas clássicas e inovadoras foram apresentadas, tanto em descrição matemática como também em resultados práticos de aplicação.

Os resultados, obtidos com dados reais de ambiente de aplicação, apresentaram métricas compreendendo todos os principais pontos de análise para uma arquitetura de processamento distribuído, em especial a proposta *edge-fog*, e seus benefícios quando comparada à solução convencional, baseada na camada de borda (no caso, no robô). Foram analisados latência, *throughput* de rede, processamento, gasto energético, memória RAM e armazenamento utilizados. A partir da melhora apresentada pela arquitetura, sua escalabilidade foi testada, mostrando resultados mais eficientes para 5 robôs em paralelo quando comparados a somente um, se este funciona por meio da arquitetura baseada na borda.

O robô final foi construído em duas versões, que se divergiam principalmente quanto ao sensor LiDAR e a aspectos construtivos. A versão mais nova demonstrou muito mais eficácia no escaneamento e qualidade de dados, após o upgrade de sensores e adaptações gerais feitas no software para absorvê-los. Houve um aumento na quantidade de dados enviados perante ao visto na primeira versão, e ainda assim a arquitetura proposta obteve bom funcionamento na distribuição dos processos. Vídeos do robô em execução estão disponíveis no Anexo B.

O Algoritmo de Octree Iterativa também sofreu reformulação e melhoria ao longo do processo de doutoramento. A nova formulação para divisão das folhas foi bem sucedida e comprovada com dados reais. A redução no número de iterações e no tempo total de execução, e a homogeneidade das nuvens de pontos nas folhas resultantes se

mostraram verdadeiras nos experimentos.

Um dos pontos levantados como trabalhos futuros disse respeito à adição do sistema ROS2. No melhor conhecimento do autor, o sistema ainda está em fases de maior adaptação a sistemas embarcados, especialmente à placa utilizada NVidia Jetson Nano. O sistema operacional Ubuntu possivelmente deveria ser substituído no processador, o que traria muitos problemas de compatibilidade com os diversos drivers, utilizados ou não diretamente pelo robô construído. Como a arquitetura funciona bem para os testes propostos, essa iniciativa ainda está em observação para um momento de aplicação da tecnologia em maior escala num ambiente como o da hidrelétrica.

Finalmente, dizendo respeito ao gargalo de rede, uma possível saída estaria na aplicação da arquitetura em uma rede 5G, cenário que ainda se inicia no Brasil. Os requisitos foram estudados e previstos como conclusão dos resultados, baseado no que temos hoje em dia nos resultados com rede Wi-fi. Com esse advento de rede, o gargalo de largura de banda e de latência ainda visto na arquitetura iria ser em teoria removido, e diversos robôs poderão ser espalhados no ambiente para reconstrução em tempo quase real como apresentado no estudo que leva em consideração o segundo cenário de aplicação na Usina Hidrelétrica (Figura 19). Como trabalho futuro, a arquitetura pode ser empregada em uma rede real 5G industrial, e novos artigos serão publicados com os resultados.

REFERÊNCIAS

1 WROBEL, A.; PLACZEK, M. Visualization systems for industrial automation systems. In: *IOP Conference Series Materials Science and Engineering*. [S.l.: s.n.], 2018. v. 400.

2 SILVA, B. P.; FERREIRA, R. A.; JR, S. C. G.; CALADO, F. A.; ANDRADE, R. M.; PORTO, M. P. On-rail solution for autonomous inspections in electrical substations. *Infrared Physics & Technology*, Elsevier, v. 90, p. 53–58, 2018.

3 YAN, L.; JIANWEI, S. Monitoring and fault diagnosis system of wind–solar hybrid power station based on zigbee and bp neural network. *Australian Journal of Mechanical Engineering*, Taylor & Francis, v. 16, n. sup1, p. 54–60, 2018.

4 LE, N. P.; GIAP, L. N. Remote monitoring system for independent power stations in rural and mountainous areas in vietnam.

5 BROSINSKY, C.; WESTERMANN, D.; KREBS, R. Recent and prospective developments in power system control centers: Adapting the digital twin technology for application in power system control centers. In: IEEE. *2018 IEEE International Energy Conference (ENERGYCON)*. [S.l.], 2018. p. 1–6.

6 SALHAOUI, M.; GUERRERO-GONZÁLEZ, A.; ARIOUA, M.; ORTIZ, F. J.; OUALKADI, A. E.; TORREGROSA, C. L. Smart industrial iot monitoring and control system based on uav and cloud computing applied to a concrete plant. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 19, n. 15, p. 3316, 2019.

7 VALERO, E.; BOSCHÉ, F.; FORSTER, A. Automatic segmentation of 3d point clouds of rubble masonry walls, and its application to building surveying, repair and maintenance. *Automation in Construction*, Elsevier, v. 96, p. 29–39, 2018.

8 PENG, H.; WANG, Q.; YANG, Y.; BI, M. Research and application of digital 3d modeling technology in substation monitoring. *DEStech Transactions on Engineering and Technology Research*, n. iceta, 2016.

9 VIDAL, V.; HONÓRIO, L.; PINTO, M.; DANTAS, M.; AGUIAR, M.; CAPRETZ, M. An edge-fog architecture for distributed 3d reconstruction and remote monitoring of a power plant site in the context of 5g. *Sensors*, MDPI, v. 22, n. 12, p. 4494, 2022.

10 VIDAL, V. F.; HONÓRIO, L. M.; DIAS, F. M.; PINTO, M. F.; CARVALHO, A. L.; MARCATO, A. L. Sensors fusion and multidimensional point cloud analysis for electrical power system inspection. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 20, n. 14, p. 4042, 2020.

11 HE, Y.; GUO, J.; ZHENG, X. From surveillance to digital twin: Challenges and recent advances of signal processing for industrial internet of things. *IEEE Signal Processing Magazine*, IEEE, v. 35, n. 5, p. 120–129, 2018.

12 PINTO, M. F.; MARCATO, A. L.; MELO, A. G.; HONÓRIO, L. M.; URDIALES, C. A framework for analyzing fog-cloud computing cooperation applied to information processing of uavs. *Wireless Communications and Mobile Computing*, Hindawi, v. 2019, 2019.

13 CHENG, M.; XIANG, D. The design and application of a track-type autonomous inspection robot for electrical distribution room. *Robotica*, Cambridge University Press, v. 38, n. 2, p. 185–206, 2020.

14 SATO, Y.; TERASAKA, Y.; UTSUGI, W.; KIKUCHI, H.; KIYOOKA, H.; TORII, T. Radiation imaging using a compact compton camera mounted on a crawler robot inside reactor buildings of fukushima daiichi nuclear power station. *Journal of Nuclear Science and Technology*, Taylor & Francis, v. 56, n. 9-10, p. 801–808, 2019.

15 SHI, W.; J.CAO; Q.ZHANG; LI, Y.; XU, L. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, IEEE, v. 3, n. 5, p. 637–646, 2016.

16 OMONIWA, B.; HUSSAIN, R.; JAVED, M. A.; BOUK, S. H.; MALIK, S. A. Fog/edge computing-based iot (feciot): Architecture, applications, and research issues. *IEEE Internet of Things Journal*, IEEE, v. 6, n. 3, p. 4118–4149, 2019.

17 CHEN, S.; ZHENG, Y.; WANG, K.; LU, W. Delay guaranteed energy-efficient computation offloading for industrial iot in fog computing. In: IEEE. *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. [S.l.], 2019. p. 1–6.

18 AAZAM, M.; ZEADALLY, S.; HARRAS, K. A. Deploying fog computing in industrial internet of things and industry 4.0. *IEEE Transactions on Industrial Informatics*, IEEE, v. 14, n. 10, p. 4674–4682, 2018.

19 GUDI, S. L. K. C.; JOHNSTON, B.; WILLIAMS, M.-A. Fog robotics: A summary, challenges and future scope. *arXiv preprint arXiv:1908.04935*, 2019.

20 BOTTA, A.; GALLO, L.; VENTRE, G. Cloud, fog, and dew robotics: Architectures for next generation applications. In: IEEE. *2019 7th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*. [S.l.], 2019. p. 16–23.

21 ACEMOGLU, A.; KRIEGLSTEIN, J.; CALDWELL, D. G.; MORA, F.; GUASTINI, L.; TRIMARCHI, M.; VINCIGUERRA, A.; CAROBBIO, A. L. C.; HYSENBELLI, J.; DELSANTO, M. et al. 5g robotic telesurgery: Remote transoral laser microsurgeries on a cadaver. *IEEE Transactions on Medical Robotics and Bionics*, IEEE, v. 2, n. 4, p. 511–518, 2020.

22 ALEKSY, M.; DAI, F.; ENAYATI, N.; ROST, P.; POCOVI, G. Utilizing 5g in industrial robotic applications. In: IEEE. *2019 7th International Conference on Future Internet of Things and Cloud (FiCloud)*. [S.l.], 2019. p. 278–284.

23 LEE, J.; LEE, K.; YOO, A.; MOON, C. Design and implementation of edge-fog-cloud system through hd map generation from lidar data of autonomous vehicles. *Electronics*, Multidisciplinary Digital Publishing Institute, v. 9, n. 12, p. 2084, 2020.

24 MELO, A. G.; PINTO, M. F.; HONÓRIO, L. M.; DIAS, F. M.; MASSON, J. E. 3d correspondence and point projection method for structures deformation analysis. *IEEE Access*, IEEE, v. 8, p. 177823–177836, 2020.

25 BALTA, H.; VELAGIC, J.; BOSSCHAERTS, W.; CUBBER, G. D.; SICILIANO, B. Fast statistical outlier removal based method for large 3d point clouds of outdoor environments. *IFAC-PapersOnLine*, Elsevier, v. 51, n. 22, p. 348–353, 2018.

26 GUO, L.; ZHANG, Y.; GUO, J.; SHI, P.; ZHAO, K. An object-oriented based 3d model for substation monitoring. In: IEEE. *2020 IEEE/IAS Industrial and Commercial Power System Asia (I&CPS Asia)*. [S.l.], 2020. p. 1469–1473.

27 ZEYBEK, M.; ŞANLIOĞLU, İ. Point cloud filtering on uav based point cloud. *Measurement*, Elsevier, v. 133, p. 99–111, 2019.

28 CAO, M.-W.; LI, L.; XIE, W.-J.; JIA, W.; LV, Z.-H.; ZHENG, L.-P.; LIU, X.-P. Parallel k nearest neighbor matching for 3d reconstruction. *IEEE Access*, IEEE, v. 7, p. 55248–55260, 2019.

29 JAKOB, J.; BUCHENAU, C.; GUTHE, M. Parallel globally consistent normal orientation of raw unorganized point clouds. In: WILEY ONLINE LIBRARY. *Computer Graphics Forum*. [S.l.], 2019. v. 38, n. 5, p. 163–173.

30 TIAN, N.; TANWANI, A. K.; CHEN, J.; MA, M.; ZHANG, R.; HUANG, B.; GOLDBERG, K.; SOJOUDI, S. A fog robotic system for dynamic visual servoing. In: IEEE. *2019 International Conference on Robotics and Automation (ICRA)*. [S.l.], 2019. p. 1982–1988.

31 SILVA, L. A. Z. da; VIDAL, V. F.; HONÓRIO, L. M.; DANTAS, M. A. R.; PINTO, M. F.; CAPRETZ, M. A heterogeneous edge-fog environment supporting digital twins for remote inspections. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 20, n. 20, p. 5296, 2020.

32 DECAMPS, M.; MEHÁUT, J.-F.; VIDAL, V.; HONORIO, L.; PIOLI, L.; DANTAS, M. A. An implementation science effort in a heterogenous edge computing platform to support a case study of a virtual scenario application. In: SPRINGER. *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. [S.l.], 2020. p. 136–147.

33 IORGA, M.; FELDMAN, L.; BARTON, R.; MARTIN, M. J.; GOREN, N. S.; MAHMOUDI, C. *NIST Special Publications - Fog Computing Conceptual Model Recommendations of the National Institute of Standards and Technology*. [S.l.], 2018. v. 500, n. 325.

34 SARKER, V.; QUERALTA, J. P.; GIA, T.; TENHUNEN, H.; WESTERLUND, T. Offloading slam for indoor mobile robots with edge-fog-cloud computing. In: IEEE. *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*. [S.l.], 2019. p. 1–6.

35 TANWANI, A. K.; MOR, N.; KUBIATOWICZ, J.; GONZALEZ, J. E.; GOLDBERG, K. A fog robotics approach to deep robot learning: Application to object recognition and grasp planning in surface decluttering. In: IEEE. *2019*

International Conference on Robotics and Automation (ICRA). [S.l.], 2019. p. 4559–4566.

36 SHAHZADI, R.; NIAZ, A.; ALI, M.; NAEEM, M.; RODRIGUES, J. J.; QAMAR, F.; ANWAR, S. M. Three tier fog networks: Enabling iot/5g for latency sensitive applications. *China Communications*, IEEE, v. 16, n. 3, p. 1–11, 2019.

37 HAYAT, S.; JUNG, R.; HELLWAGNER, H.; BETTSTETTER, C.; EMINI, D.; SCHNIEDERS, D. Edge computing in 5g for drone navigation: What to offload? *IEEE Robotics and Automation Letters*, IEEE, v. 6, n. 2, p. 2571–2578, 2021.

38 BRAVO-ARRABAL, J.; TOSCANO-MORENO, M.; FERNANDEZ-LOZANO, J.; MANDOW, A.; GOMEZ-RUIZ, J. A.; GARCÍA-CEREZO, A. The internet of cooperative agents architecture (x-ioca) for robots, hybrid sensor networks, and mec centers in complex environments: A search and rescue case study. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 21, n. 23, p. 7843, 2021.

39 SHIN, S.; KIM, J.; MOON, C. Road dynamic object mapping system based on edge-fog-cloud computing. *Electronics*, Multidisciplinary Digital Publishing Institute, v. 10, n. 22, p. 2825, 2021.

40 OPEN3D octree Point Cloud division. 2021. <http://www.open3d.org/docs/latest/tutorial/geometry/octree.html>.

41 ROTH, S. D. Ray casting for modeling solids. *Computer graphics and image processing*, Elsevier, v. 18, n. 2, p. 109–144, 1982.

42 ARTEAGA, A. O.; SCOTT, D.; BOEHM, J. Initial investigation of a low-cost automotive lidar system. In: COPERNICUS GMBH. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. [S.l.], 2019. v. 42, p. 233–240.

43 GLENNIE, C.; HARTZELL, P. Accuracy assessment and calibration of low-cost autonomous lidar sensors. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, Copernicus GmbH, v. 43, p. 371–376, 2020.

44 THE Point Cloud Library - Removing outliers using a StatisticalOutlierRemoval filter. 2020. <https://pcl.readthedocs.io/projects/tutorials/en/master/statistical_outlier.html>.

45 THE Point Cloud Library - Estimating Surface Normals in a PointCloud. 2020. <https://pcl.readthedocs.io/en/latest/normal_estimation.html>.

46 HABIBI, P.; FARHOUDI, M.; KAZEMIAN, S.; KHORSANDI, S.; LEON-GARCIA, A. Fog computing: a comprehensive architectural survey. *IEEE Access*, IEEE, v. 8, p. 69105–69133, 2020.

47 BADAT, L.; VIDAL, V.; PIOLI, L.; MEHAUT, J.-F.; HONORIO, L.; DANTAS, M. A. R. An iiot edge environment as a main support to a 3d reconstruction virtualization application. In: SBC. *Anais do XVIII Workshop em Clouds e Aplicações*. [S.l.], 2020. p. 1–12.

48 ARKIAN, H. R.; DIYANAT, A.; POURKHALILI, A. Mist: Fog-based data analytics scheme with cost-efficient resource provisioning for iot crowdsensing applications. *Journal of Network and Computer Applications*, Elsevier, v. 82, p. 152–165, 2017.

49 VASCONCELLOS, V.; CARDOSO, B. B.; MARTINS, K. A.; MACEDO, A. F. de; CECCHETTI, B. F.; MARTINS, M. A. I. On the application of 5g to energy distribution companies: a brazilian case study. In: IEEE. *2021 IEEE Latin-American Conference on Communications (LATINCOM)*. [S.l.], 2021. p. 1–5.

50 MUZAFFAR, R.; RAFFELSBERGER, C.; FAKHREDDINE, A.; LUQUE, J. L.; EMINI, D.; BETTSTETTER, C. First experiments with a 5g-connected drone. In: *Proceedings of the 6th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*. [S.l.: s.n.], 2020. p. 1–5.

APÊNDICE A – PUBLICAÇÕES RELACIONADAS AO TRABALHO

Neste anexo estão listados os trabalhos publicados frutos dos resultados da pesquisa realizada no âmbito deste trabalho de doutorado.

A.1 ARTIGOS DE CONFERÊNCIA

BADAT, LEYA ; Vidal, Vinicius ; PIOLI, LAÉRCIO ; MEHAUT, JEAN-FRANCOIS ; HONORIO, LEONARDO ; DANTAS, MARIO ANTONIO RIBEIRO . An IIoT Edge Environment as a Main Support to a 3D Reconstruction Virtualization Application. In: XVIII Workshop em Clouds e Aplicações, 2020, Brasil. Anais do XVIII Workshop em Clouds e Aplicações (WCGA 2020). p. 1.

Abstract: This paper describes an experimental research work, which was conducted to gather different types of cameras and computer node, as IIoT (Industrial IoT) devices, to produce support to a digital transformation with a 3D reconstruction virtualization for a real engineering application. The computational approach considered was a heterogeneous edge computing environment. In this environment, different cameras and computer node architectures, collaborate as heterogeneous IIoT processing elements, to provide a better, and as fast as possible, images to a virtualization project. The challenge and complexity related to software packages orchestration, from these IIoT devices, are also reported.

A.2 CAPÍTULOS DE LIVRO

Decamps, Marceau ; Meháut, Jean-Francois ; Vidal, Vinicius ; HONORIO, LEONARDO ; PIOLI, LAÉRCIO ; Dantas, Mario A. R. . An Implementation Science Effort in a Heterogenous Edge Computing Platform to Support a Case Study of a Virtual Scenario Application. Lecture Notes in Networks and Systems. 1ed.: Springer International Publishing, 2021, v. 158, p. 136-147.

Abstract: IoT devices are pillars for the Industry 4.0 software applications. However, clustering these edge nodes are interesting open challenges in several dimensions, because of mandatory integration of diverse hardware and software packages. Different type of industrial cameras and a supercomputer node to support 3D reconstructions is not a trivial approach, especially considering aspects of the IoT software engineering. In this paper, we present a research, which could be classified as an implementation science effort. The target is a heterogenous edge computing platform, utilized to support a real case study of an electrical engineering field application. This application is characterized by a 3D virtual reconstruction paradigm for the hydropower project. Our results indicate interesting aspects related to implementation science and challenges found in the composition and operationalization of this heterogenous edge platform.

A.3 TRABALHOS ACEITOS E PUBLICADOS EM PERIÓDICOS INTERNACIONAIS

A.3.1 PUBLICAÇÕES RELACIONADAS

VIDAL, VINICIUS F.; HONÓRIO, LEONARDO M. ; DIAS, FELIPE M. ; PINTO, MILENA F. ; CARVALHO, ALEXANDRE L. ; MARCATO, ANDRE L. M. . Sensors Fusion and Multidimensional Point Cloud Analysis for Electrical Power System Inspection. SENSORS, v. 20, p. 4042, 2020.

Abstract: Thermal inspection is a powerful tool that enables the diagnosis of several components at its early stages. One critical aspect that influences thermal inspection outputs is the infrared reflection from external sources. This situation may change the readings, demanding that an expert correctly define the camera position, which is a time consuming and expensive operation. To mitigate this problem, this work proposes an autonomous system capable of identifying infrared reflections by filtering and fusing data obtained from both stereo and thermal cameras. The process starts by acquiring readings from multiples Observation Points (OPs) where, at each OP, the system processes the 3D point cloud and thermal image by fusing them together. The result is a dense point cloud where each point has its spatial position and temperature. Considering that each point's information is acquired from multiple poses, it is possible to generate a temperature profile of each spatial point and filter undesirable readings caused by interference and other phenomena. To deploy and test this approach, a Directional Robotic System (DRS) is mounted over a traditional human-operated service vehicle. In that way, the DRS autonomously tracks and inspects any desirable equipment as the service vehicle passes them by. To demonstrate the results, this work presents the algorithm workflow, a proof of concept, and a real application result, showing improved performance in real-life conditions.

Avaliação CAPES: Qualis A2

Fator de impacto: 3.847

SILVA, LUIZ A. Z. DA ; VIDAL, VINICIUS F. ; HONÓRIO, LEONARDO M. ; DANTAS, MÁRIO A. R. ; PINTO, MILENA FARIA ; CAPRETZ, MIRIAM . A Heterogeneous Edge-Fog Environment Supporting Digital Twins for Remote Inspections. SENSORS, v. 20, p. 5296, 2020.

Abstract: The increase in the development of digital twins brings several advantages to inspection and maintenance, but also new challenges. Digital models capable of representing real equipment for full remote inspection demand the synchronization, integration, and fusion of several sensors and methodologies such as stereo vision, monocular Simultaneous Localization and Mapping (SLAM), laser and RGB-D camera readings, texture analysis, filters, thermal, and multi-spectral images. This multidimensional information makes it possible to have a full understanding of given equipment, enabling remote diagnosis. To solve this problem, the present work uses an edge-fog-cloud architecture running over a publisher-subscriber communication framework to optimize the computational costs and throughput. In this approach, each process is embedded in an edge node responsible for prepossessing a given amount of data that optimizes the trade-off of processing capabilities and throughput delays. All information is integrated with different levels of fog nodes and a cloud server to maximize performance. To demonstrate this proposal, a real-time 3D reconstruction problem using moving cameras is shown. In this scenario, a stereo and RDB-D cameras run over edge nodes, filtering, and prepossessing the initial data. Furthermore, the point cloud and image registration, odometry, and filtering run over fog clusters. A cloud server is responsible for texturing and processing the final results. This approach enables us to optimize the time lag between data acquisition and operator visualization, and it is easily scalable if new sensors and algorithms must be added. The experimental results will demonstrate precision by comparing the results with ground-truth data, scalability by adding further readings and performance.

Avaliação CAPES: Qualis A2

Fator de impacto: 3.847

A.3.2 PUBLICAÇÕES ORIGINADAS DA TESE

Vidal, V. F., Honorio, L. M., Pinto, M. F., Dantas, M. A., Aguiar, M. J., & Capretz, M. (2022). An edge-fog architecture for distributed 3D reconstruction. Future Generation Computer Systems.

Abstract: Nowadays, numerous facilities that provide essential services operate with little or even no human supervision. In this context, electrical power substations can be entirely autonomous or remotely operated, and may be difficult to access. This scenario increases the need of remote monitoring for predictive maintenance and problem analysis by using imaging and 3D reconstruction of the environment. 3D reconstruction processes are concentrated in academic and commercial applications, but they are non scalable and fail to provide remote real time data analysis. This research proposes a novel color 3D scanner architecture environment for remote real time multiple sensor processing and reconstruction. The goal is to present a more efficient system based upon scalability and low latency applications, where multiple sensors can be added without losing the overall processing capability. For this purpose, this study approaches this problem in two ways. First, it improves 3D reconstruction algorithms by enhancing individual performance. Second, to optimize the entire system, it distributes the running processes in individual layers, interconnected by an edge–fog architecture. This architecture enables the use of multiple devices by optimizing payload distribution, latency, and throughput in the network. Unlike previous studies, the results present a thorough analysis of architecture efficiency when multiple sensors are operating in parallel instead of the traditional centralized architecture. Finally, this work provides the basis for real-time remote presence applications.

Avaliação CAPES: Qualis A1

Fator de impacto: 7.307

Vidal, V., Honório, L., Pinto, M., Dantas, M., Aguiar, M., & Capretz, M. (2022). An Edge-Fog Architecture for Distributed 3D Reconstruction and Remote Monitoring of a Power Plant Site in the Context of 5G. Sensors, 22(12), 4494.

Abstract: It is well known that power plants worldwide present access to difficult and hazardous environments, which may cause harm to on-site employees. The remote and autonomous operations in such places are currently increasing with the aid of technology improvements in communications and processing hardware. Virtual and augmented reality provide applications for crew training and remote monitoring, which also rely on 3D environment reconstruction techniques with near real-time requirements for environment inspection. Nowadays, most techniques rely on offline data processing, heavy computation algorithms, or mobile robots, which can be dangerous in confined environments. Other solutions rely on robots, edge computing, and post-processing algorithms, constraining scalability, and near real-time requirements. This work uses an edge-fog computing architecture for data and processing offload applied to a 3D reconstruction problem, where the robots are at the edge and computer nodes at the fog. The sequential processes are parallelized and layered, leading to a highly scalable approach. The architecture is analyzed against a traditional edge computing approach. Both are implemented in our scanning robots mounted in a real power plant. The 5G network application is presented along with a brief discussion on how this technology can benefit and allow the overall distributed processing. Unlike other works, we present real data for more than one proposed robot working in parallel on site, exploring hardware processing capabilities and the local Wi-Fi network characteristics. We also conclude with the required scenario for the remote monitoring to take place with a private 5G network.

Avaliação CAPES: Qualis A2

Fator de impacto: 3.847

APÊNDICE B – VÍDEOS ILUSTRATIVOS

Os *links* a seguir ilustram ambas as versões desenvolvidas do robô em funcionamento, seus resultados, e como o processamento de dados se dá no ambiente virtual.

- Primeira versão:<https://www.youtube.com/watch?v=4wrU9og-eng&list=PLxg9pywnxQpW
- Segunda versão: <https://www.youtube.com/watch?v=ih2824om6oI&list=PLxg9pywnxQpW index=2>
- Resultados em geral: <https://www.youtube.com/watch?v=0BxJtsuHLpo&list= PLxg9pywnxQpWviJFq4EfQp9Y2dGogp5SY&index=3>

Como uma extensão das funcionalidades do robô, as imagens obtidas foram utilizadas para criação de imagens 360 graus, e a navegação no ambiente reconstruído se dá de forma virtual:

- Tour virtual pela subestação de Santos Dumont (primeira versão): <https:// www.youtube.com/watch?v=fh4281Bt9oE&list=PLxg9pywnxQpW2g5cxG2-vIXX-3Pjd5b3W
- Tour virtual na pista lateral da subestação: <https://www.youtube.com/watch? v=1nt06sksOww&list=PLxg9pywnxQpW2g5cxG2-vIXX-3Pjd5b3W&index=2>
- Tour virtual na sala de comando: <https://www.youtube.com/watch?v=dkvL9eb-m4E& list=PLxg9pywnxQpW2g5cxG2-vIXX-3Pjd5b3W&index=3>