Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Estudo Sobre Redução de Redundância Geométrica de Nuvens de Pontos de Sistemas Automotivos

Dayanne Fernandes da Cunha

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Ricardo Lopes de Queiroz

Brasília
2021

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Estudo Sobre Redução de Redundância Geométrica de Nuvens de Pontos de Sistemas Automotivos

Dayanne Fernandes da Cunha

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Ricardo Lopes de Queiroz (Orientador)
CIC/UnB

Prof. Dr. Camilo Chang Dórea
CIC/UnB

Prof. Dr. Díbio Leandro Borges
CIC/UnB

Prof. Dr. Marcelo Grandi Mandelli
Coordenador do Bacharelado em Ciência da Computação

Brasília, 27 de Maio de 2021

Dedicatória

Dedico este trabalho a todos que me auxiliaram na minha jornada na Universidade de Brasília (UnB). Aos professores que além de ensinar também trouxeram uma energia e motivação essencial para que os dias fossem mais fáceis de serem trilhados. Professoras(es) como a profa. Cláudia Melo, que praticamente me adotou nos últimos semestres e me encorajou a continuar no curso; prof. Díbio Borges, que trouxe excelentes desafios e animava os alunos com gestos simples; profa. Carla Castanho que foi a professora que me ensinou o ABC da programação; prof. Ricardo Queiroz que me orientou nesta última caminhada na universidade, e com muita paciência me guiou até este momento; prof. André Drummond que me deu assistência quando eu mais precisava em momentos de perda de sanidade; profa. Alba Cristina que em meus anos de *IEEE* foi uma excelente companheira de organização de eventos e projetos de extensão.

Também dedico aos professores de outros departamentos que colaboraram neste caminho, como a profa. Mariana Bernardes, prof. Geovany Borges e prof. Antônio Padilha que me concederam a oportunidade de participar de um dos projetos mais impactantes da minha vida, o projeto *Clara*. Dedico este espaço também ao Prof. Rafael Timóteo e aos pesquisadores Lucas Martins, Francisco Lopes e Fábio Mendonça que me proporcionaram uma visão otimista da academia e me auxiliaram na publicação do meu primeiro artigo em conferência.

Dedico também aos meus amados amigos, Yurick Hauschild, Pedro Abreu, Julianna Brandão, Teogenes Moura e Andreza Bona. Sem eles nada disso seria possível, me trouxeram muito companheirismo, amor e suporte.

> We are just misguided ghosts.
> Travelin' endlessly.
>
> *Hayley Williams*

Agradecimentos

Agradeço imensamente a todo suporte que a UnB proveu-me, ela foi uma casa, uma mãe, um abrigo que tive por muitos anos. Nela ganhei conhecimento, comida, uma condição de vida e também descobri meus limites, minhas paixões, meus ódios e minha garra. Não foi uma trajetora fácil, passei por milhares de dificuldades físicas, mentais e financeiras. Mas sem esta trajetória e o apoio da instituição eu não teria nada do que conquistei até hoje. Considero que, além desta carta de agradecimento, também consegui devolver minha gratidão à instituição e à comunidade, através de inúmeras contribuições aos projetos de pesquisa e extensão que colaborei.

Agradeço ao Decanato de Assuntos Comunitários (DAC), Diretoria de Desenvolvimento Social (DDS), Fundação Universidade de Brasília (FUB), Restaurante Universitário (RU), Centro De Atendimento E Estudos Psicológicos (CAEP), Departamento de Ciência da Computação (CIC) e Biblioteca Central (BCE) pelos anos de apoio, excelente infraestrutura e auxílio psicológico e financeiro. E por fim, mas não menos importante, agradeço imensamente à minha revisora de texto, Bianca Alencar Vellasco.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

Este estudo propõe avaliar um método de redução de redundância geométrica, também denominado de redundância temporal ou inter-quadros, de nuvens de pontos adquiridas dinamicamente por um sensor *Light Detection And Ranging (LiDAR)*, sendo este sensor comumente utilizado em sistemas automotivos e autônomos. Por meio de experimentos com a base de dados *Kitti*, foi possível avaliar os benefícios do método proposto na unificação de nuvem de pontos em comparação ao método *Iterative Closest Point (ICP) Point-to-Plane (Po2Pl)*.

Palavras-chave: LiDAR, Nuvem de Pontos, 3D, Redundância Geométrica

Abstract

This study proposes to evaluate a method to reduce geometric redundancy, also called temporal or inter-frame redundancy, of point clouds acquired dynamically by a *Light Detection And Ranging (LiDAR)* sensor, which is commonly used in automotive and autonomous systems. Through experiments with the *Kitti* dataset it was possible to assess the benefits of the proposed point cloud registration method compared to the *Iterative Closest Point (ICP) Point-to-Plane (Po2Pl).*

Keywords: LiDAR, Point Cloud, 3D, Geometric Redundancy

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Equações

Lista de Abreviaturas e Siglas

ADAS Advanced Driver Assistance System.

AEB Automatic Emergency Braking.

BCE Biblioteca Central.

bit binary digit.

CAEP Centro De Atendimento E Estudos Psicológicos.

CIC Departamento de Ciência da Computação.

CMY Cyan, Magenta, Yellow.

CMYK Cyan, Magenta, Yellow, Black.

DAC Decanato de Assuntos Comunitários.

DDS Diretoria de Desenvolvimento Social.

FUB Fundação Universidade de Brasília.

HSI Hue, Saturation, Intensity.

ICP Iterative Closest Point.

IFC International Finance Corporation.

IIP Iterative Innovation Point.

IMU Inertial Measurement Unit.

k-NN k-Nearest Neighbors.

LiDAR Light Detection And Ranging.

LTS Long-Term Support.

PCL Point Cloud Library.

Po2Pl Point-to-Plane.

Po2Po Point-to-Point.

RAHT Region-Adaptive Hierarchical Transform.

RGB Red, Green, Blue.

ROI Região de Interesse.

RU Restaurante Universitário.

SAE Society of Automotive Engineers.

SE Special Euclidean Group.

SO Special Orthogonal Group.

UnB Universidade de Brasília.

WHO World Health Organization.

ZMV Zero Motion Vector.

Capítulo 1

Introdução

1.1 Motivação

De acordo com uma pesquisa realizada pela *McKinsey & Company* em dezembro de 2020 [12], o risco de contaminação entre as pessoas se tornou o principal fator de determinação na escolha de um veículo de transporte. Sendo assim, discussões sobre veículos próprios, *e-hailings*[1], e veículos autônomos se tornaram alvo das mídias e do mercado de investimentos.

Ainda nesta pesquisa lançada pela *McKinsey & Company*, é mencionado que o interesse por carros próprios foi maior que o por carros compartilhados e *e-hailings*, justamente pelo alto medo de contato com as pessoas e eventuais infecções. Com isso em mente, muitas indústrias que em até 2019 tiveram um alto valor de investimento, acabaram sendo negativamente impactadas pela pandemia.

A pesquisa [13] mostra que, entre os anos 2010 e 2019, os *e-hailings* ficaram no topo da lista de investimentos dentro do setor de mobilidade com 56.2 bilhões investidos em *start-ups*[2]. Podemos então concluir que estas empresas terão que se adaptar rapidamente nos próximos anos para que o consumidor volte a confiar nestes meios de transporte.

Devido ao confinamento completo ou parcial aplicado em muitos países como meio de evitar a expansão do vírus *Covid-19*, o setor de logística sofreu um grande impacto. Este setor se viu diante de muita demanda tanto de pessoas físicas quanto jurídicas. Muitos comerciantes tiveram que adaptar seu plano de negócio e focar em entregas ao em vez de compras em lojas físicas, e as pessoas cada vez mais foram reduzindo a preferência em selecionar um produto por meio de visitas físicas, optando por compras virtuais. Devido

[1] Veículos de transporte privados requisitados por um aplicativo baseado em sua geolocalização, e.g. *Uber*, *Cabify*, etc.

[2] Empresa emergente.

a esta troca de hábitos no consumo e nos planos de negócios, o uso de meios de transporte para mercadorias também foi algo muito discutido e requisitado.

Em 2020, o *International Finance Corporation (IFC)* do *World Bank Group* publicou um estudo sobre o impacto do *Covid-19* na logística mundial [14], um impacto que suscitou níveis de adaptação drásticos para muitas empresas, que envolveram, por exemplo, a criação de novos protocolos de segurança, meios alternativos de transporte, montagem de estratégias eficazes para lidar com a alta capacidade nos caminhões de entregas, etc. Devido a este impacto e a crescente demanda logística, a previsão é de que, a longo prazo, robôs, drones e veículos autônomos sejam cada vez mais utilizados para reduzir a exposição dos prestadores de serviços de logística e abrir um leque maior de opções de transporte seguro e rastreável.

Além do impacto mundial causado pela pandemia nos meios de transporte e logística, muitas empresas investem em veículos autônomos de forma geral, pois estes veículos podem impactar positivamente outros setores. No artigo [15], publicado em 2016, são apresentadas diferentes perspectivas do impacto de veículos autônomos, como uma melhor infraestrutura urbana, maior acessibilidade para as pessoas com deficiência física, redução de acidentes em ruas e estradas, melhora no fluxo de trânsito, entre muitos outros benefícios.

No estudo [16], publicado em 2015, foi feita uma análise que identificava que grandes companhias de carro como a *Audi*, *GM* e *Mercedes-Benz* iriam investir na produção de carros parcialmente ou completamente autônomos até a metade de 2020. Agora, em 2021, já temos uma atualização dessa previsão: empresas como a *Audi* [17] e a *Tesla* [18] conseguiram alcançar a produção de carros parcialmente autônomos e continuam a investir para que a automação total seja conquistada.

Em abril de 2021, foi lançado um manual atualizado pela *Society of Automotive Engineers (SAE)* a respeito dos níveis dos carros autônomos [19], a definição apresentada traz seis níveis, sendo eles:

- Nível 0: **Sem automação no veículo**. Completamente controlado por um ser humano.
- Nível 1: **Assistência na direção**. Com alguns avisos e controle mínimo no carro, como por exemplo alguns carros que previnem colisões com freios emergenciais automáticos, conhecidos por *Automatic Emergency Braking (AEB)*.
- Nível 2: **Automação parcial**. Neste nível já é possível considerar o veículo como um *Advanced Driver Assistance System (ADAS)*, um termo muito conhecido no mercado e nas pesquisas acadêmicas. O veículo é capaz de controlar tanto aceleração/desaceleração como também a direção.

- Nível 3: **Automação condicional**. Neste momento, do ponto de vista tecnológico há um salto enorme, dado que o carro é capaz de detectar seu ambiente ao redor e tomar as decisões parciais com base nesta detecção.
- Nível 4: **Alto nível de automação**. Neste nível, além do carro captar seu ambiente ao redor e os comportamentos diários de trajeto e do motorista, ele também pode interferir se algo der errado, como por exemplo se o motorista dormir no volante. Atualmente, este nível só é possível em determinadas regiões pré-determinadas chamadas de *geofencing*[1].
- Nível 5: **Automação total**. Neste nível não é necessário nenhuma intervenção humana, o carro é capaz de detectar tudo ao seu redor e reagir da maneira mais adequada de acordo com cada circunstância.

De acordo com uma pesquisa de 2018 da *World Health Organization (WHO)* [20], aproximadamente 1.35 milhão de pessoas morrem anualmente por morte no trânsito. Já *Marchant, Gary* e *Lindor, Rachel* [21] apresentam uma discussão a respeito de carros autônomos e o sistema de responsabilidade a nível jurídico e ético global. A discussão nesta pesquisa realizada por eles fica em torno da questão da responsabilidade em acidentes de trânsito em que carros autônomos estão envolvidos. Até o Nível 2 de sistemas autônomos, a tendência é responsabilizar totalmente o motorista em caso de acidentes, e só a partir do Nível 3 esta responsabilidade recairia sobre os fabricantes daquele automóvel.

Para evitar possíveis colisões e prever melhor o trajeto é também de muito interesse dos produtores e usuários de carros autônomos de todos os níveis o preparo de mapas eficientes para esta tecnologia. Existem múltiplas ferramentas que provém mapas para carros sem automação, como por exemplo o *Google Maps* [22] e o *Waze* [23], mas nenhum destes mapas provê suporte para comunicação com um sistema autônomo.

Veículos autônomos possuem um potencial grandioso para o futuro da população, pois afeta diretamente muitos setores essenciais tanto de forma positiva quanto negativa a depender do contexto. Mas um fato é: essas tecnologias já estão entre nós, e a cada dia mais motivos se acumulam para que estes veículos protagonizem um papel de grande responsabilidade, como o controle da mobilidade e da logística sem a interferência humana. Sendo assim, a produção de tecnologias eficientes para serem implantadas nesses carros devem ser meticulosamente pensadas, para que elas se tornem de fato um meio de auxiliar o ser humano.

Em vista desta motivação econômica do setor automobilístico e de mobilidade urbana na confecção de mapas para sistemas autônomos, este presente estudo irá buscar métricas

[1]Perímetro geográfico para determinar uma área de interesse, também chamado de "geocerca".

de redução de redundância de dados e irá demonstrar o uso destas nuvens reduzidas através da formação de mapas tridimensionais.

1.2 Definição do Problema

A partir do Nível 3 de automação veicular, o carro já começa a coletar informações do espaço ao seu redor, e para isso existem múltiplas tecnologias que auxiliam nesta captação de dados, como por exemplo o *Light Detection And Ranging (LiDAR)*. Este tipo de sensor geralmente capta dados em ciclos de milissegundos e pode conter entre 80 mil a mais de 1 milhão de pontos geométricos que caracterizam as superfícies dos objetos e cenários ao redor. Quanto maior a resolução captada, melhor a qualidade do dado, porém, o ganho de qualidade implica em outros problemas, como a alta taxa de transmissão destes dados e o alto custo computacional para computá-los.

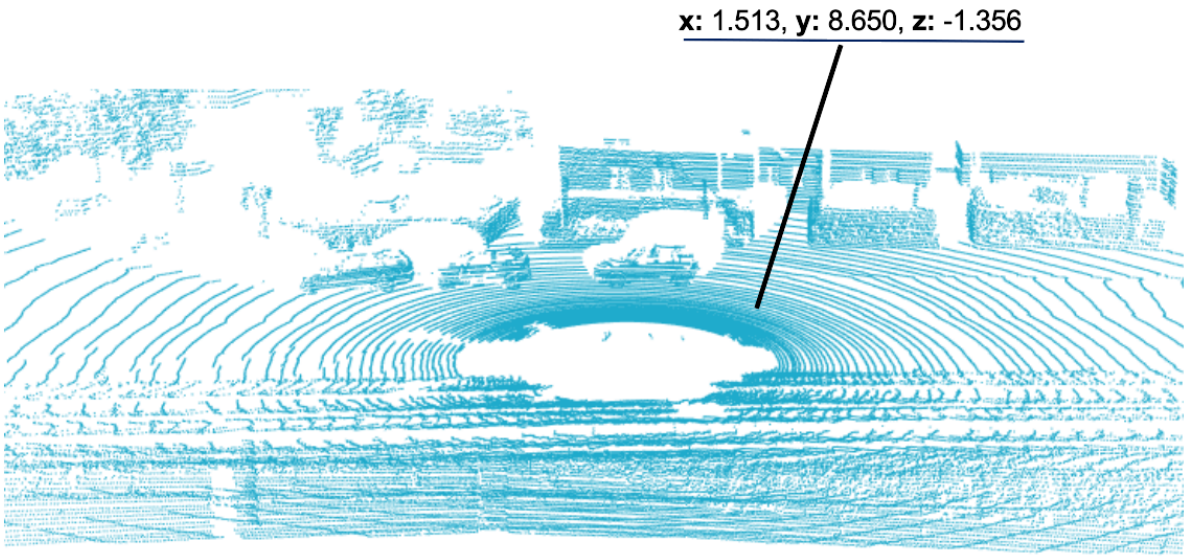

Figura 1.1: Nuvem de pontos captada por um sensor *LiDAR*.

Na Figura 1.1 vemos um dos tipos de dados captados a partir do Nível 3 de automação. Esse dado é utilizado para entender o ambiente ao redor do carro, onde é possível analisar outros veículos, pessoas, prédios, e outros objetos que possam interferir nas ações do veículo. Esse dado é denominado nuvem de pontos. Na Figura 1.1 imagem existem exatamente 120574 pontos flutuantes. O sensor que capta este tipo de dado geralmente faz a uma frequência de 10 Hz, ou seja, capta 10 nuvens de pontos por segundo. Abaixo na Equação 1.1 podemos então compreender o problema em termos numéricos:

$$
\begin{aligned}
f &= 10 * 60 = 600 \\
qp &= 120574 \\
b &= 32 \\
d &= 3
\end{aligned}
\tag{1.1}
$$

$$
\begin{aligned}
t &= qp * b * d * f \\
&= 120574 * 32 * 3 * 600 \\
&= 6945062400\ bits \\
&\approx 868.1328\ MB/minuto
\end{aligned}
$$

.

Dado uma frequência de 10 nuvens de pontos por segundo, e um minuto contendo 60 segundos, temos então **f** a frequência de nuvem de pontos por minuto, **qp** a quantidade de pontos flutuantes da nuvem em questão, **b** a quantidade de bits que um ponto flutuante utiliza computacionalmente e **d** a dimensão do plano de nuvem. Então, a partir da equação acima teremos aproximadamente 868 MB por minuto, o que se aproxima de quase 1 GB por minuto. Portanto, vemos que este é um dos tipos de dados que o carro autônomo coleta durante o trajeto, e se esse valor se propagar nos dados dos outros sensores então estamos diante de um problema de alta quantidade de dados a serem transmitidos, computados e/ou armazenados.

Além do *LiDAR*, os veículos autônomos são equipados por outros sensores, como câmeras, sistemas de navegação inercial, etc. A junção destes dados proporciona a montagem de um mapa hiper realista, de alta resolução e rico em detalhes, para que a automação se torne mais eficaz. Quanto maior a quantidade e qualidade dos dados, maior a certeza para as ações e reações tomadas pelo veículo.

Em [24], foi realizado uma prospecção tecnológica de futuros desafios que a indústria automobilística encontrará na criação de carros autônomos seguros, confiáveis, eficientes e com um custo benefício razoável. Neste estudo, é mencionado que há uma preocupação com o processamento dos dados captados pelos carros autônomos e sua velocidade de transmissão em tempo real para situações de identificação de cenários e objetos vizinhos.

1.3 Proposta

O que se propõe neste trabalho é apresentar uma métrica de redução de redundância geométrica dos dados captados por um sensor *LiDAR* acoplado a um carro. Como dito

acima, na Seção 1.2, este sensor produz uma quantidade enorme de informação ao longo do trajeto, e como o período de captação é curto, o perímetro que o carro percorre entre um ciclo de captação e outro também não é grande, e isso gera dados redundantes do cenário ao redor.

Este problema de redundância de pontos captados entre um ciclo e outro é denominado por **redundância temporal**, algo também chamado de **redundância inter-quadros**. A fim de montar um mapa, unificando estes pontos captados pelo *LiDAR*, sem muita informação repetida, é então proposto um método que identifica os pontos redundantes a cada iteração, remove-os, e constrói o mapa apenas com os pontos novos que são detectados a cada iteração.

Para a experimentação deste método, será utilizado o conjunto de dados abertos *Kitti*[8]. Serão utilizados dados de dois sensores deste conjunto, as nuvens de pontos captadas por um *LiDAR Velodyne HDL-64E* e dados de geolocalização e movimento do carro captados pelo sistema de navegação inercial *GPS/IMU OxTS RT 3003*.

1.4 Organização do Trabalho

O presente trabalho foi organizado em cinco capítulos, incluindo este introdutório, para uma melhor compressão do tema, do problema e da proposta. O Capítulo 2 foca em apresentar a revisão bibliográfica de trabalhos relacionados e os conceitos fundamentais a respeito de sistemas autônomos e seus sensores, nuvem de pontos e suas representações e métodos de unificação das nuvens de pontos para construção de um mapa.

O Capítulo 3 traz a descrição da arquitetura do método, assim como os módulos, a estrutura dos códigos e os experimentos executados. E para encerrar, os Capítulos 4 e 5 apresentam a análise dos dados, discussão dos resultados, e a conclusão do estudo.

Capítulo 2

Fundamentação Teórica

2.1 Introdução

Para compreender como a redundância geométrica em nuvem de pontos de sistemas automotivos pode acontecer, é preciso compreender a definição formal das nuvens de pontos, que será mostrada na Subseção 2.2.2, e suas representações por meio de matrizes e árvores, como no exemplo da *k-D Tree*, apresentada em seguida na Subseção 2.2.3. A descrição das tecnologias que captam esses tipos de dados é também crucial para compreender o problema, portanto, a Seção 2.3 apresenta detalhes a respeito do *LiDAR* e *GPS/IMU*.

Após a compreensão dos dados, os fundamentos sobre tipos de redundâncias são apresentados na Seção 2.4, e na Seção 2.5 é exposto como o sistema de coordenadas interfere na construção do mapa com as nuvens de pontos. Na sequência, na Seção 2.6, ocorre o momento de detalhamento sobre os métodos conhecidos na literatura para unificar as nuvens. Por fim, na Seção 2.7, são apresentados trabalhos relacionados à redução de redundância e compressão de nuvens de pontos.

2.2 Representações Visuais

2.2.1 Quadros 2D

Uma imagem bidimensional pode ser representada de duas formas: *rasterizada* e através de vetores. A primeira representação, que geralmente é a mais utilizada, utiliza um mapa de *binary digit (bit)* para montar uma unidade denominada por *pixel*. Para apresentação e controle da estrutura deste mapa, é utilizado uma matriz conforme apresentado na Equação 2.1 [1]. Sendo $M, N \in \mathbb{N}$ a representação da resolução daquela imagem, e cada $f(x = i, y = j) = a_{ij}, i, j \in \mathbb{N}$ representa a cor do *pixel* de acordo com o sistema de cores utilizado.

$$F(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & ... & f(0,N-1) \\ f(1,0) & f(0,1) & ... & f(0,N-1) \\ ... & ... & ... & ... \\ f(M-1,0) & f(M-1,1) & ... & f(M-1,N-1) \end{bmatrix} \tag{2.1}$$

Para representar as cores de uma imagem, existem alguns sistemas conhecidos de cores: o usual *Red, Green, Blue (RGB)* apresentado na Figura 2.1, o *Cyan, Magenta, Yellow (CMY)*, *Cyan, Magenta, Yellow, Black (CMYK)*, e o *Hue, Saturation, Intensity (HSI)*.

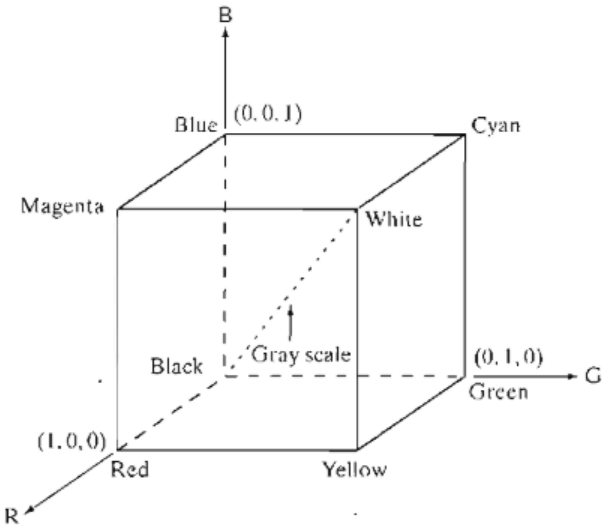

Figura 2.1: Esquemático (x, y, z) do sistema de cores *RGB*. Imagem extraída do livro [1].

2.2.2 Quadros 3D

Para representar imagens/objetos tridimensionais também é utilizado uma matriz, como apresentado na Equação 2.2 [25]. Onde, ao invés de $F(x, y)$, agora temos $F(x, y, z)$ para representar os elementos na imagem. Cada elemento $f(x, y, z)$ pode apresentar mais atributos a respeito daquele ponto no espaço além do atributo comum de cor.

$$F(x,y,z=0)=\begin{bmatrix} f(0,0,0) & f(0,1,0) & ... & f(0,N-1,0) \\ f(1,0,0) & f(0,1,0) & ... & f(0,N-1,0) \\ ... & ... & ... & ... \\ f(M-1,0,0) & f(M-1,1,0) & ... & f(M-1,N-1,0) \end{bmatrix}$$

$$F(x,y,z=1)=\begin{bmatrix} f(0,0,1) & f(0,1,1) & ... & f(0,N-1,1) \\ f(1,0,1) & f(0,1,1) & ... & f(0,N-1,1) \\ ... & ... & ... & ... \\ f(M-1,0,1) & f(M-1,1,1) & ... & f(M-1,N-1,1) \end{bmatrix}$$

$$F(x,y,z=k)=\begin{bmatrix} f(0,0,k) & f(0,1,k) & ... & f(0,N-1,k) \\ f(1,0,k) & f(0,1,k) & ... & f(0,N-1,k) \\ ... & ... & ... & ... \\ f(M-1,0,k) & f(M-1,1,k) & ... & f(M-1,N-1,k) \end{bmatrix}$$

...

$$F(x,y,z=l)=\begin{bmatrix} f(0,0,l) & f(0,1,l) & ... & f(0,N-1,L-1) \\ f(1,0,l) & f(0,1,l) & ... & f(0,N-1,L-1) \\ ... & ... & ... & ... \\ f(M-1,0,L-1) & f(M-1,1,L-1) & ... & f(M-1,N-1,L-1) \end{bmatrix} \tag{2.2}$$

O equivalente dos *pixels* para imagens tridimensionais são os *voxels*, e ele pode ser representado em uma célula tal que $f(x=i,y=j,z=k)=a_{ijk}, i,j,k \in \mathbb{N}$, e $M,N,L \in \mathbb{N}$ sendo a_{ijk} o valor da densidade daquele *voxel*.

Além dos *voxels*, existem também outros meios de representar a imagem tridimensional, como por exemplo as chamadas *nuvens de pontos*. Neste caso, cada $a_{ijk}=(x_t,y_t,z_t)$, tal que $(x_t,y_t,z_t) \in \mathbb{R}^3$. Essa coleção de pontos geralmente está no mesmo sistema de coordenadas, como apresentado na Figura 2.2, e além de representar o ponto geométrico em um plano, a_{ijk} também pode conter outros atributos, como cor, reflectância do objeto, vetor normal, etc.

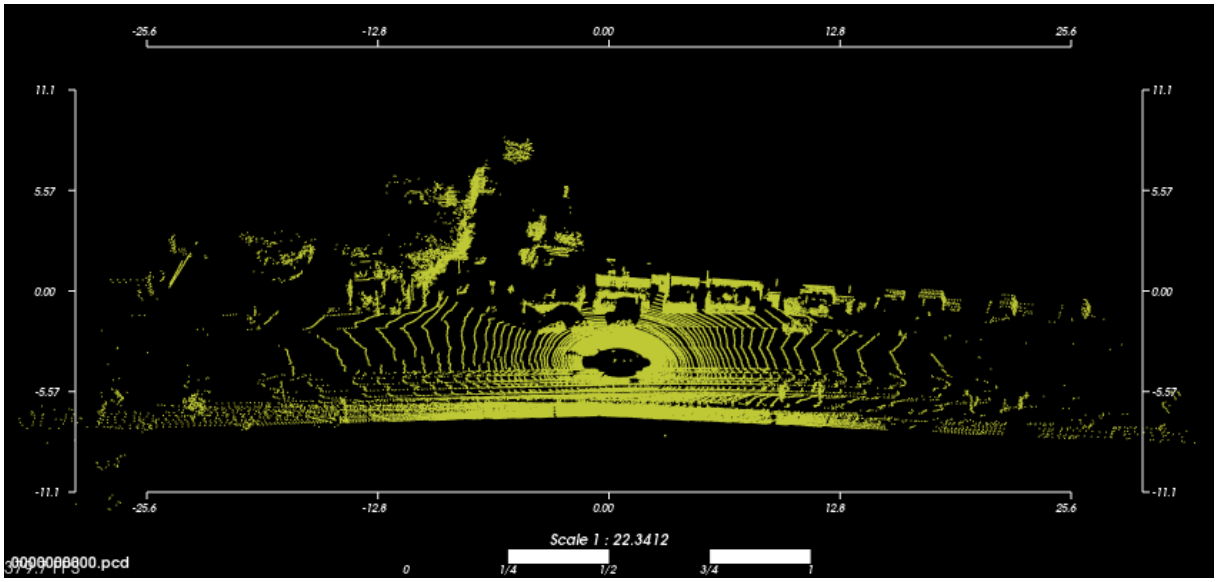

Figura 2.2: Exemplo de uma nuvem de pontos. Imagem renderizada através da ferramenta *Point Cloud Library (PCL)* [2].

Existem dois tipos de modalidades de uso das nuvens de pontos, sendo elas: a modalidade de uso destas nuvens de forma unitária/estática, que compõe alguma representação de forma isolada; e as nuvens de pontos dinâmicas, esparsas na dimensão temporal. Na Figura 2.3 temos exemplos de nuvens de pontos que representam diferentes tipos de superfícies, como na figura **(a)** apresentando o corpo de uma pessoa, na figura **(b)** um mapa e na figura **(c)** um objeto.

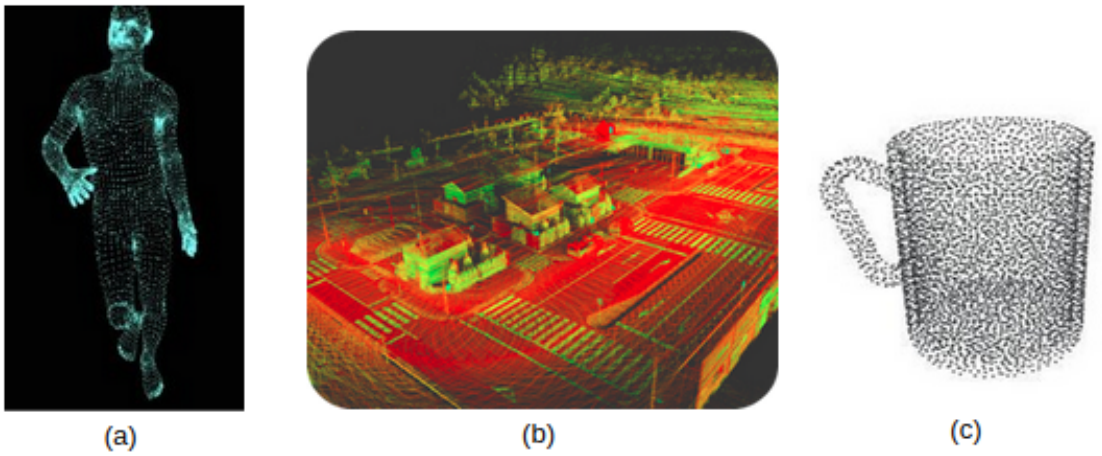

Figura 2.3: Nuvens de pontos representando diferentes superfícies. Figura **(a)** da fonte [3], figura **(b)** da fonte [4] e figura **(c)** da fonte [5].

2.2.3 K-D Tree

Existem várias formas de representar uma nuvem de pontos além de matrizes e vetores, sendo possível também representá-las computacionalmente, através de estruturas de dados denominadas árvores, a exemplo da *QuadTree*, *Octree* e a *k-D Tree*. Cada estrutura de dados possui suas vantagens e desvantagens, e neste trabalho focaremos na descrição dos benefícios da estrutura *k-D Tree*, criada pelo cientista da computação *Jon Bentley*, em 1975.

A *k-D Tree* é uma abreviação para árvore binária de busca com k dimensões, de acordo com o *D. E. Knuth* [26]. A árvore binária pode ser definida por uma tupla (T, r), sendo T a árvore e r o nó raiz da árvore, e sendo T composta de apenas um nó raiz r e cada nó possui no máximo três conexões, ou seja, duas conexões com os nós filhos e uma conexão com o nó pai. No caso da *k-D Tree*, e no contexto deste trabalho o $k = 3$ devido ao manuseio de representações tridimensionais.

Esta árvore possui benefícios tanto de ordem de complexidade de tempo no custo de construção dos nós e de busca, quanto de ordem de uso de memória como é apresentado na Tabela 2.1.

Método	**Tempo**	**Memória**
Criação	O(nlogn)	O(n)
Busca	O(logn)	O(n)

Tabela 2.1: Complexidade de tempo e memória em operações com a *k-D Tree* em caso médio de acordo com [9].

Esta estrutura de dados é comumente utilizada para buscas de nós na árvore e de seus vizinhos, a exemplo do algoritmo *k-Nearest Neighbors (k-NN)*. Esses tipos de buscas demandam um custo computacional altíssimo, por exemplo, para realizar a busca de um ponto em uma nuvem de pontos de um quadro k com tamanho N_k, utilizando métodos de busca por força bruta, a ordem do cálculo é de $O(N_k)$, e se quisermos calcular a distância de todos os pontos a todos os pontos em dois quadros de nuvens de pontos, a ordem se torna $O(N_i N_k)$, sendo i o primeiro quadro e k o segundo quadro na comparação, sendo N_i o tamanho total do primeiro quadro. Se formos utilizar a *k-D Tree* para os mesmos problemas, teremos $O(logN_k)$ e $O(N_i logN_k)$, respectivamente [27].

2.3 Sensores para Veículos Autônomos

2.3.1 LiDAR

O *Light Detection And Ranging (LiDAR)* é uma tecnologia óptica que emite pulsos de raios infra vermelhos e calcula a distância a uma superfície de um objeto através do tempo e velocidade de retorno do pulso refletido. Existem várias empresas que produzem essa tecnologia, e ela pode ser utilizada em vários segmentos, como construção de mapas, estruturação de prédios, detecção de profundidade de rios, lagos e mares, agricultura, entre outros.

Sensores *LiDAR* podem ser utilizados em diferentes locais e tipos de veículos a fim de captar dados relevantes do ambiente, podendo ser acoplados em carros, caminhões, drones, postes, robôs de entrega, robôs de limpeza, etc. Em carros e caminhões, podem ser utilizados para reconhecimento do percurso. A tecnologia vinculada em locais estáticos nas ruas auxilia na mensuração da densidade populacional local. Em robôs e veículos autônomos em geral, o *Light Detection And Ranging (LiDAR)* é essencial para evitar colisões. Em drones, é comumente utilizado para mapear profundezas de vegetações, e detecção de anomalias, como inundações em locais remotos e danos à sistemas de distribuição elétrica.

Uma das marcas conhecidas tanto na literatura quanto no mercado é a *Velodyne* [28]. Essa empresa foi criada em 1983, e vem produzindo, desde 2004, soluções de sensores para o avanço de carros autônomos. Em 2010, a *Alphabet*, antiga *Google*, começa a testar carros autônomos nas ruas de São Francisco com sensores da *Velodyne*, especificamente o *HDL-64E*.

O sensor *HD-64E* [29] da *Velodyne* é um LiDAR tridimensional de alta definição com captação de dados em tempo real e campo de visão de 360º. Esse sensor é importante pelo fato de estar repetidamente presente na captação de muitos conjuntos de dados abertos utilizados em pesquisa de sistema autônomos. Um dos conjuntos de dados de grande notoriedade acadêmica que utilizou este sensor para elaborar os cenários de testes foi o *KITTI* [8].

2.3.2 Sistema de Navegação Inercial

Os sistemas de navegação inercial, geralmente identificados por *GPS/IMU* são comumente utilizados em veículos autônomos a fim de captar dados de geolocalização, como a latitude, longitude e altitude, dados da velocidade, aceleração e taxas angulares do veículo. Este dispositivo disponibiliza estes dados em dois sistemas de coordenadas, um vinculado ao

carro (x, y, z) e outro mapeado através de um plano tangente à superfície da Terra (f, l, u) [30].

2.4 Redundância de Dados

Existem dois tipos de redundância, a espacial, também denominada por intra-quadro, letra *a* da Figura 2.4, e a temporal, denominada por inter-quadro e apresentada na letra *b* da Figura 2.4. No contexto do intra-quadro, a busca da redundância é feita no mesmo quadro/imagem e no inter-quadro a redundância é avaliada entre múltiplos quadros [6].

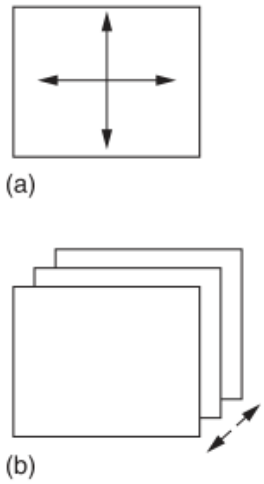

Figura 2.4: Tipos de redundância, espacial e temporal. Imagem editada e extraída da fonte [6].

Neste trabalho, o que denominamos por redundância geométrica se refere à redundância inter-quadro. Para avaliar esta redundância, é utilizado o cálculo de distância euclidiana dos pontos geométricos entre os dois quadros subsequentes. O cálculo é então definido na Equação 2.3.

$$\begin{aligned} p_1 &= (x_1, y_1, z_1) \\ p_2 &= (x_2, y_2, z_2) \\ d(p_1, p_2) &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \end{aligned} \tag{2.3}$$

Para evitar que este cálculo seja feito de todos os pontos a todos os pontos independente da localização, é então montado um algoritmo para checar um limite mínimo. O algoritmo é apresentado a seguir:

Algorithm 1 $euclidean\_distance(p_1, p_2)$

1: **return** $\sqrt{(p_2.x - p_1.x)^2 + (p_2.y - p_1.y)^2 + (p_2.z - p_1.z)^2}$

Algorithm 2 $check\_threshold(p_1, p_2, THRESHOLD)$

1: **if** $((p_2.x + THRESHOLD > p_1.x)$ && $(p_1.x > p_2.x - THRESHOLD))$ && $((p_2.y + THRESHOLD > p_1.y)$ && $(p_1.y > p_2.y - THRESHOLD))$ && $((p_2.z + THRESHOLD > p_1.z)$ && $(p_1.z > p_2.z - THRESHOLD))$ **then**
2: **return** TRUE
3: **end if**
4: **return** FALSE

Algorithm 3 $calc\_euclidean\_distance(quadro_1, quadro_2, THRESHOLD)$

1: **for** $ponto_i$ *in* $quadro_1$ **do**
2: **for** $ponto_j$ *in* $quadro_2$ **do**
3: **if** $check\_threshold(ponto_i, ponto_j, THRESHOLD)$ **then**
4: $euclidean\_distance(ponto_i, ponto_j)$
5: **end if**
6: **end for**
7: **end for**

Esse mesmo procedimento, como dito anteriormente, pode ter um alto custo computacional. Portanto, neste trabalho, este cálculo foi feito através de um *k-D Tree* utilizando o algoritmo de *radiusSearch* da biblioteca *nanoflann* [31]. Este algoritmo encontra N vizinhos de um nó baseado na busca pelo raio, ou seja, um perímetro limiar, e ao utilizar a métrica *L2_Simple_Adaptor* ele performa essa busca calculando a distância Euclidiana, conforme explicado nos algoritmos acima.

2.5 Sistemas de Coordenadas

Para representar pontos e vetores, duas abordagens são utilizadas, uma sintética e uma analítica. Na abordagem sintética, eles são representados como entidades geométricas, e na analítica, por meio de coordenadas ou equações, em que a manipulação ocorre por meio de técnicas algébricas [7]. Nas próximas Subseções, iremos utilizar a abordagem analítica para descrever as transformadas homogêneas da movimentação do objeto rígido em análise, um carro.

Na Figura 2.5, dois sistemas de coordenadas distintos são apresentados, um plano expresso por coordenadas cartesianas para representar os dados coletados por um *Light*

Detection And Ranging (LiDAR), e outro plano cartesiano para representar as coordenadas do carro. Neste trabalho, será necessário realizar relações geométricas do carro em relação ao plano do *LiDAR*, portanto, vamos descrever nas Subseções 2.5.1 e 2.5.2 como podemos criar e usar essas relações.

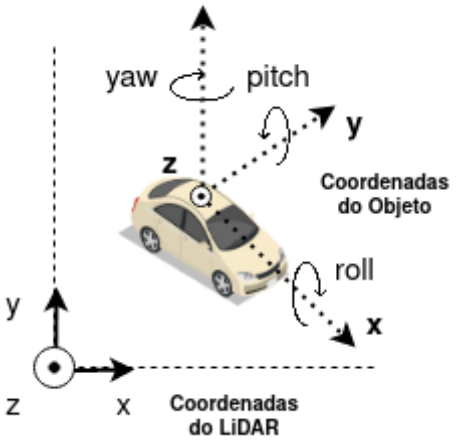

Figura 2.5: Relação entre o sistema de coordenadas do LiDAR com as coordenadas do sistema de navegação inercial *GPS/IMU* do carro.

2.5.1 Ângulos de Euler

Um objeto rígido em um plano tridimensional possui três graus de liberdade independentes, e esses graus podem ser representados pelos famosos ângulos de Euler. Sejam dois quadros de coordenadas fixos $R^0 = o_0x_0y_0z_0$ e $R^1 = o_1x_1y_1z_1$, com suas origens em o_0 e o_1, respectivamente. Podemos especificar a orientação de R^1 em relação ao quadro R^0 utilizando três ângulos (ϕ, θ, ψ), conforme é mostrado na Figura 2.6. Cada ângulo é obtido por três rotações sucessivas em cada eixo. Existem doze combinações de rotações, e neste trabalho vamos utilizar a sequência X-Y-Z para a criação os ângulos *pitch*, *yaw* e *roll*, representados por ϕ, θ, e ψ, respectivamente.

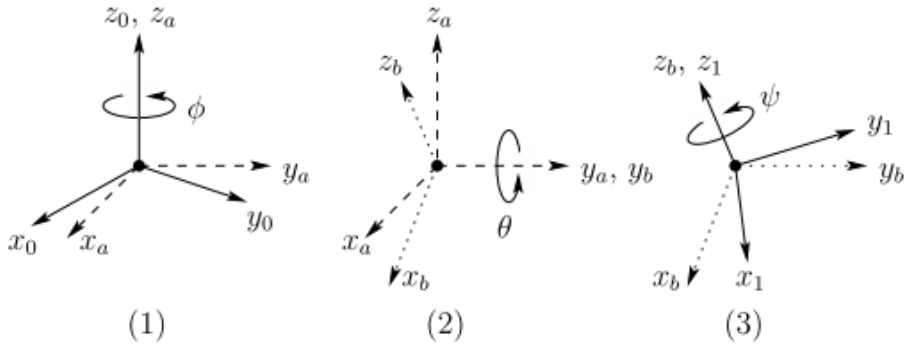

Figura 2.6: Representação dos ângulos de Euler, imagem do livro [7].

2.5.2 Rotação e Translação

Um objeto tridimensional pode ser rotacionado através de três eixos ortogonais conforme descrito na Subseção 2.5.1, sendo eles o eixo lateral, eixo vertical e o eixo longitudinal. Na literatura, esses eixos são conhecidos por *pitch*, *yaw* e *roll* ou por ϕ, θ, e ψ, respectivamente. As matrizes desses eixos estão representadas nas Equações 2.4 a 2.6, sendo r_z, r_y e r_x os ângulos *pitch*, *yaw* e *roll* em radianos [32].

$$R_z(r_z) = \begin{bmatrix} cos(r_z) & -sin(r_z) & 0 \\ sin(r_z) & cos(r_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$R_y(r_y) = \begin{bmatrix} cos(r_y) & 0 & sin(r_y) \\ 0 & 1 & 0 \\ -sin(r_y) & 0 & cos(r_y) \end{bmatrix} \tag{2.5}$$

$$R_x(r_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(r_x) & -sin(r_x) \\ 0 & sin(r_x) & cos(r_x) \end{bmatrix} \tag{2.6}$$

$$R(r_z, r_y, r_x) = R_z(r_z)R_y(r_y)R_x(r_x) =$$
$$\begin{bmatrix} cos(r_y)cos(r_y) & cos(r_z)sin(r_y)sin(r_x) - sin(r_z)cos(r_x) & cos(r_z)sin(r_y)cos(r_x) + sin(r_z)sin(r_x) \\ sin(r_z)cos(r_y) & sin(r_z)sin(r_y)sin(r_x) + cos(r_z)cos(r_x) & sin(r_z)sin(r_y)cos(r_x) - cos(r_z)sin(r_x) \\ -sin(r_y) & cos(r_y)sin(r_x) & cos(r_y)cos(r_x) \end{bmatrix} \tag{2.7}$$

Um movimento rígido é composto de uma translação e uma rotação, onde podemos definir o ato de transladar como uma simples soma de vetores, e.g. seja a translação definida por $t = (x_t, y_t, z_t)$, tal que $t \in \mathbb{R}^3$, se o carro estava em um ponto $p_0 = (x_0, y_0, z_0)$, então no final teremos $p_1 = (x_0 + x_t, y_0 + y_t, z_0 + z_t)$.

Sendo o grupo ortogonal definido pela sigla *Special Orthogonal Group (SO)*, e $SO(3)$ o grupo composto por três dimensões, a definição da matriz de transformação homogênea de um objeto rígido é apresentada na Equação 2.8, sendo $R \in SO(3)$ e $t \in \mathbb{R}^3$. O grupo de todos os movimentos rígidos é conhecido na literatura por *Special Euclidean Group (SE)*, e vamos utilizá-lo como $SE(3) = \mathbb{R}^3 \times SO(3)$, onde $H \in SE(3)$ [7].

$$H = \begin{bmatrix} rotação & translação \\ perspectiva & fator\ de\ escala \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{2.8}$$

2.5.3 Projeção de Mercator

Mercator é uma projeção cilíndrica sob o globo terrestre criada pelo cartógrafo e matemático *Gerhard Mercator*. Na projeção de *Mercator*, a escala apresentada na Equação 2.9 é calculada a partir dos meridianos, utilizando a latitude como referência para manter a escala igual por todas as direções [33]. A partir do cálculo da escala e da constante do raio da Terra, é possível então criar a Equação 2.10 para obter um ponto (x, y) em coordenada cartesiana a partir da latitude/longitude de um objeto.

$$\begin{aligned} pi &= 3.14159 \\ scale &= cos\left(\frac{(latitude * pi)}{180}\right) \end{aligned} \tag{2.9}$$

$$
\begin{aligned}
&earth\_radius = 6378137 \\
&x = \frac{scale * longitude * pi * earth\_radius}{180} \\
&y = scale * earth\_radius * log\left(tan\left(\frac{(90 + latitude) * pi}{360}\right)\right)
\end{aligned}
\tag{2.10}
$$

2.6 Unificação de Nuvens de Pontos

A unificação das nuvens de pontos na literatura é denominada por *registration*. Este ato é utilizado a fim de formar mapas através destes pontos geométricos dinâmicos esparsos temporalmente. Existem algumas técnicas conhecidas para unificá-los, como por exemplo o *Iterative Closest Point (ICP)*.

O *ICP* possui duas categorias, *Point-to-Point (Po2Po)* e *Point-to-Plane (Po2Pl)*. Existem múltiplas variações de ambos, mas vamos definir a variação [10] de *Po2Po* na Equação 2.11, e de [11] de *Po2Pl* na Equação 2.12. Sendo $\mathbf{K} = \{(p, q)|p \in P, q \in Q\}$ a nuvem de pontos alinhada, onde P representa o quadro no instante de tempo t e Q no instante $t + 1$. T representa a matriz de transformação encontrada para alinhar P e Q. E n_p apresentada na Equação 2.12 representa a normal de p.

$$
E(T) = \sum_{(p,q)\in\mathbf{K}} ||p - Tq||^2 \tag{2.11}
$$

$$
E(T) = \sum_{(p,q)\in\mathbf{K}} ((p - Tq).n_p)^2 \tag{2.12}
$$

Assim como o *ICP*, neste trabalho é apresentado uma proposta para realizar a unificação de nuvens de pontos utilizando o método da *k-D Tree* com o cálculo da distância Euclidiana para detectar pontos de inovação a cada iteração.

2.7 Trabalhos Relacionados

No artigo [34], Souto e Queiroz aplicam a transformada *Region-Adaptive Hierarchical Transform (RAHT)* com predições inter-quadros de baixo custo computacional, sem uso do *Zero Motion Vector (ZMV)*, como proposta alternativa para compressão de atributos de nuvens de pontos dinâmicas. A técnica preditiva inter-quadros utiliza a representação

da nuvem de pontos por *octrees* e uso do cálculo de menor distância euclidiana entre os *voxels* para estimar o quadro que será codificado.

O uso de dados geométricos representados por três dimensões em tempo real tem sido utilizado cada vez mais para diferentes situações, como por exemplo em videoconferências, jogos, e arquitetura de prédios e instalações. O artigo [35] mostra um método de compressão, sem perdas, de nuvem de pontos geométricas, denominado **intra-frame**, para auxiliar no envio desses dados tridimensionais em tempo real. Este método utiliza a estrutura de dados **octree** para comprimir os pontos baseados em sua entropia. O resultado da aplicação deste método proposto, denominado por **P(LZW)**, para comprimir a nuvem de pontos, demonstrou uma melhora de 29% em média comparando com a compressão de outros métodos, como MPEG, Draco, OR, AC, P(AC) e LZW.

Outro estudo que busca resolver o problema de compressão de dados para envio em tempo real é o [36]. Nele é apresentado um método sem perda de compressão de dados redundantes espaciais e temporais. Muitos estudos de compressão utilizam a representação destes dados por uma **octree**, porém eles desenvolveram uma modificação desta árvore ao ponto que foi possível detectar e comprimir diferenças espaciais em nuvem de pontos não estruturadas e temporalmente adjacentes.

Já o estudo [37] utilizou nuvem de pontos captadas pelo sensor *LIDAR*, a fim de fazer um mapeamento de um local utilizando uma sequência de nuvem de pontos temporais. Foi proposto um método de mapeamento e redução de redundância para gerar um mapa preciso. Para registro dos mapas temporais foi utilizado o método **ICP**, e para redução de redundância foi utilizado um algoritmo que calcula a distância de todos os pontos para todos os pontos e elimina os pontos muito próximos uns aos outros.

O software apresentado em [38] apresenta como remover objetos dinâmicos em uma nuvem de pontos. O sistema identifica estes objetos através da ocupação dos pontos em seus determinados *voxels*, elimina-os e cria uma nova nuvem de pontos somente com objetos estáticos. Para isso, foi utilizado múltiplas nuvens de pontos esparsas e temporais com a presença de objetos dinâmicos e estáticos.

O artigo [39] mostra outro método de compressão e representação de nuvem de pontos. Nele, foi utilizada a **quad-tree**, e não a **octree**, como nos estudos mencionados acima. Com essa representação, foi possível reduzir a quantidade de nós para representar uma mesma nuvem de pontos.

Considerando os estudos mencionados acima, é possível dizer então que muitos métodos de simplificação e compressão de nuvem de pontos, espaciais e temporalmente adjacentes, envolvem representá-la por uma estrutura de dados mais eficiente e que facilite identificar possíveis redundâncias. Logo, este trabalho também se propõe a desenvolver mais um método de redução de redundância temporal dessas nuvens através da estrutura

de dados denominada por *k-D Tree*.

Capítulo 3

Desenvolvimento

3.1 Introdução

Este capítulo trata da base de dados utilizada neste trabalho, detalhada na Seção 3.2, e também apresenta informações a respeito de como e quando foram obtidos e disponibilizados estes dados. Já na Seção 3.3, toda a arquitetura do trabalho é detalhada, desde a obtenção dos dados, conversões de formatos, transformações dos dados, cálculos aplicados e nuvens reduzidas como saída do sistema. E na conclusão deste capítulo, temos a Seção 3.6, que explica qual cenário do *Kitti* foi utilizado e o código desenvolvido.

3.2 Base de Dados: Kitti

Existem alguns conjuntos de dados usados em pesquisas sobre carros autônomos, como o *Kitti* [8], explicado com mais detalhes nesta seção, o *ApolloScape* [40], criado em 2019 pela empresa *Baidu*. A famosa produtora de carros *Audi* também lançou um conjunto de dados abertos em 2020, denominado por *A2D2* [17].

O conjunto de dados *Kitti* [8] foi gerado pelo Instituto de Tecnologia *Karlsruhe* em parceria com o Instituto tecnológico *Toyota* de Chicago, em 2013. Utilizando dez dispositivos acoplados a um carro, foi possível coletar cerca de 180 GB de dados para estudo e pesquisa de sistemas autônomos. Os dispositivos utilizados estão apresentados nas Figuras 3.1 a 3.2 e estão detalhados abaixo:

- 1 Sistema de navegação inercial *(GPS/IMU): OxTS RT 3003*;
- 1 *LiDAR Velodyne HDL-64E*;
- 2 Câmeras em escala em cinza, *Point Grey Flea 2 (FL2-14S3M-C), 1/2"Sony ICX267 CCD*, de 1.4 Megapixels, *global shutter*;

- 2 Câmeras coloridas, *Point Grey Flea 2 (FL2-14S3C-C), 1/2"Sony ICX267 CCD*, de 1.4 Megapixels, *global shutter*;
- 4 Lentes Varifocal, de 4-8 mm, abertura de 90º, abertura vertical de Região de Interesse (ROI) de 35º, da *Edmund Optics NT59-917*.

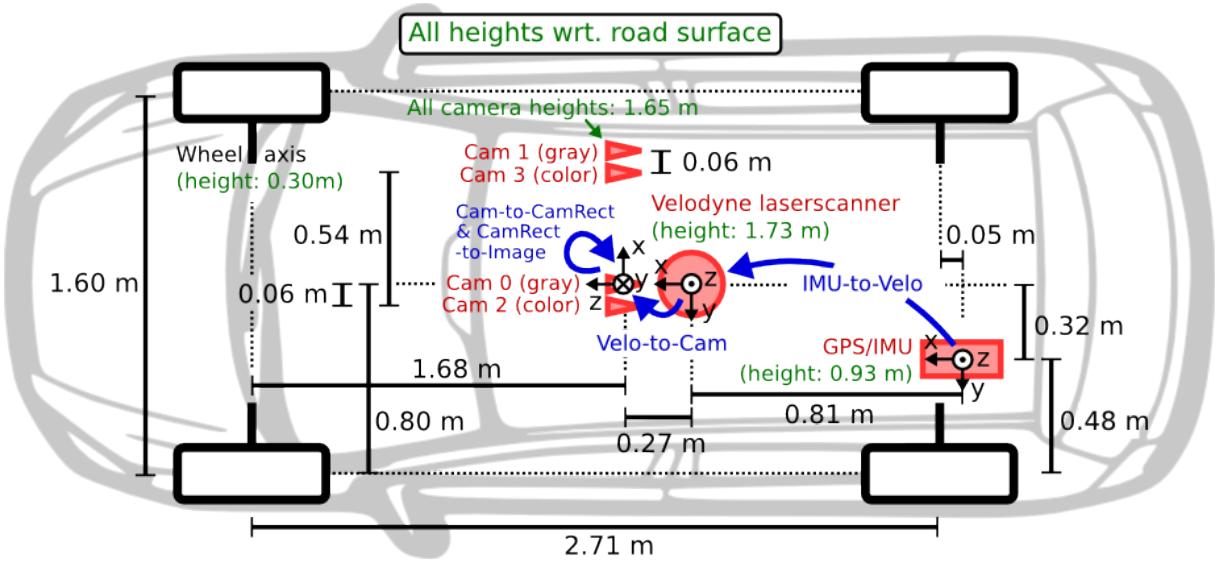

Figura 3.1: Carro e sensores utilizados na montagem do conjunto de dados *Kitti* [8].

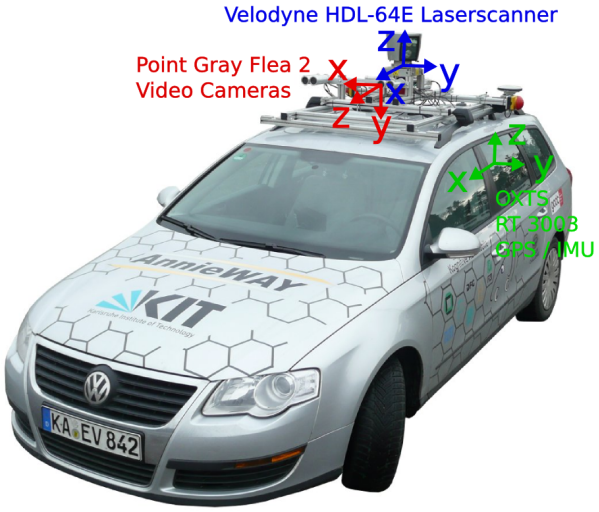

Figura 3.2: Orientação dos sistemas de coordenadas dos sensores do carro utilizado na montagem do conjunto de dados *Kitti* [8].

Existem seis categorias de cenários onde foram coletadas as nuvens de pontos, sendo elas: cidade, área residencial, estrada, campus, pessoas e cenário de calibração. Cada

categoria está organizada por uma coleção de dados coletados em um dia, mês, ano e experimento específico. Um experimento pode conter tamanhos variados, indo de poucos *MBs* até 20 GB. Todos os experimentos do *Kitti* possuem informações de todos os sensores do carro listado acima, e também oferecem uma versão sem sincronia entre alguns sensores e outra versão sincronizando os dados do *LiDAR*, *GPS/IMU* e os das imagens das câmeras de acordo com cada *timestamp*.

Além dos dados básicos como *GPS/IMU*, coordenadas geométricas e reflectância do *LiDAR*, velocidade e aceleração, o conjunto de dados também é composto por dados semânticos de acordo com a cena capturada, informando a localização e quantidade de carros, vans, caminhões, pedestres, pessoas sentadas, ciclistas, bondinhos e outros formatos gerais. Os tipos de dados captados pelo *GPS/IMU* do *Kitti* que são utilizados neste trabalho são apresentados na Tabela 3.1.

Dados	**Formato**	**Descrição**
lat	(deg) grau	latitude
lon	(deg) grau	longitude
alt	(m) metros	altitude
roll	(rad) radianos	ângulo roll, entre -pi e +pi
pitch	(rad) radianos	ângulo pitch, entre -pi/2 e +pi/2
yaw	(rad) radianos	ângulo yaw, entre -pi e +pi

Tabela 3.1: Dados *GPS/IMU* do *Kitti* utilizados neste trabalho.

O sensor LiDAR, acoplado no carro usado no projeto *Kitti*, captura nuvem de pontos, também chamada de quadros na literatura. A cada giro do *LiDAR*, é captada uma nuvem de ponto. São capturados 10 quadros por segundo, ou seja, a cada 100 milissegundos são capturados aproximadamente 100 mil pontos por ciclo. As nuvens de pontos do *Kitti* vem em formato *bin* na versão sincronizada, sendo cada linha composta por $(x, y, z, i) \in \mathbb{R}^3$. Sendo (x, y, z) as coordenadas em espaço tridimensional, e i a intensidade da reflectância do sensor luminoso *Velodyne HDL-64E* sobre a superfície do objeto escaneado. Informações mais detalhadas a respeito deste sensor foram expostas na Seção 2.3 deste trabalho.

3.3 Arquitetura

Para o teste do método aqui proposto de redução da redundância geométrica, foi montado um sistema que, a partir de dados já pré coletados de um LiDAR, e um sistema de navegação inercial *(GPS/IMU)*, fosse capaz de gerar nuvens de pontos com tamanho de

arquivo reduzido e também uma quantidade menor de pontos geométricos, sem perder a utilidade destas nuvens de pontos na formação de um mapa.

O sistema foi então dividido em duas partes, conforme apresentado na Figura 3.3, sendo a arquitetura do primeiro módulo apresentada com mais detalhes na Figura 3.4, e a do segundo na Figura 3.5.

A caixa em verde na Figura 3.3 representa os dados de entrada já descritos na Seção 3.2. A caixa em azul representa os dados gerados após a execução do primeiro módulo do sistema, que é explicado com maiores detalhes na Subseção 3.4, que tem como foco adaptar as nuvens de pontos para coordenadas globais utilizando os dados do sistema de navegação inercial do *KITTI*. Passada a execução do primeiro módulo, onde teremos as nuvens de pontos em coordenadas globais, é possível então captar os pontos redundantes.

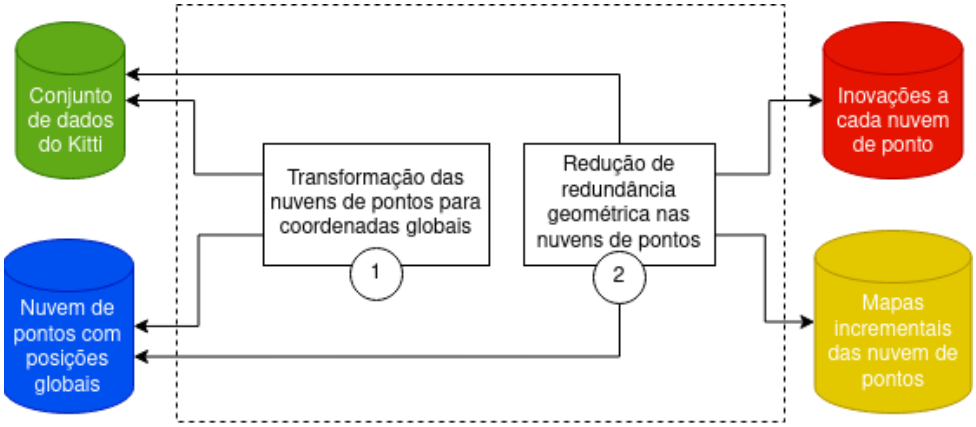

Figura 3.3: Arquitetura geral do sistema de redução de redundância geométrica, com componentes de entrada e saída.

3.4 Nuvem de Pontos em Coordenadas Globais

Para transformar as nuvens de pontos em coordenadas globais, foi preciso utilizar dados do sistema de navegação inercial *GPS/IMU*, especificamente dados de latitude, longitude, altitude, *yaw*, *pitch* e *roll* do carro utilizado no experimento. Utilizando a projeção de *Mercator*, detalhada na Subseção 2.5.3 e a matriz de transformação R, detalhada na Subseção 2.5.2, foi possível transformar as coordenadas locais das nuvens de pontos do *Kitti* em globais, os resultados destas transformações serão apresentados no Capítulo 4.

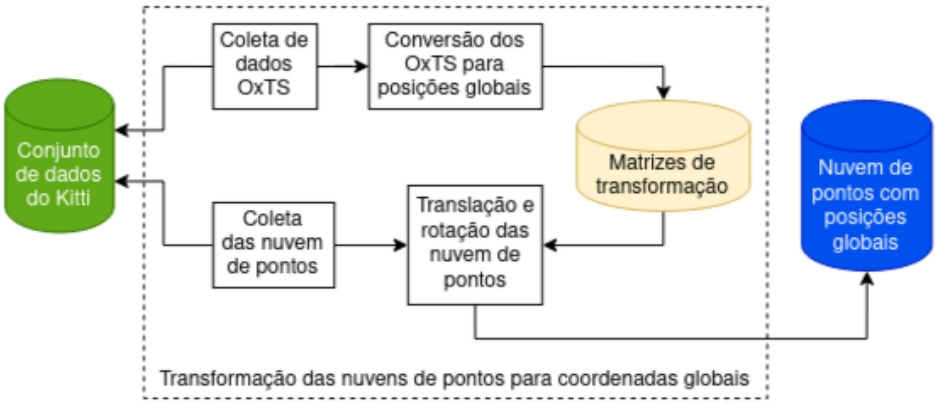

Figura 3.4: Detalhes arquiteturais da primeira parte do sistema de redução de redundância geométrica, com componentes de entrada e saída.

3.5 Redução Geométrica das Nuvens de Pontos

A lógica proposta nesse módulo é, dado que o movimento entre dois quadros dinamicamente adquiridos é pequeno, então seria interessante que a quantidade de informações geométricas novas que podem ser coletadas seja também reduzida. Cada quadro do conjunto de dados *Kitti* possui aproximadamente 120,000 pontos geométricos, então vamos avaliar se entre um quadro $\{(x_t, y_t, z_t) \in \mathbb{R}^3\}$ e o próximo quadro $\{(x_{t+1}, y_{t+1}, z_{t+1}) \in \mathbb{R}^3\}$ seria possível remover pontos já conhecidos e focar nas inovações da nova captura.

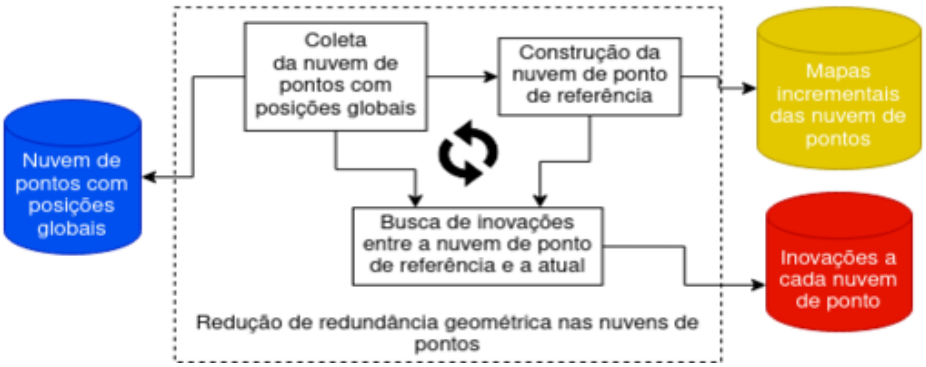

Figura 3.5: Detalhes arquiteturais da segunda parte do sistema de redução de redundância geométrica, com componentes de entrada e saída.

Dado que as nuvens de pontos já estão em coordenadas globais após a execução do módulo descrito na Subseção 3.4, então seguindo a arquitetura mostrada na Figura 3.5, são

então construída as nuvens de ponto de referência e as de inovações. Para que seja possível fazer a unificação dos quadros adquiridos dinamicamente com uma quantidade reduzida de pontos, é necessário segmentar as novas informações a cada novo quadro. Então, como podemos ver na Figura 3.6, a cada nuvem de pontos, é gerada uma nuvem de referência para que no próximo quadro tenhamos uma base a ser comparada e segmentar apenas pontos geométricos que antes não tinham sido mapeados.

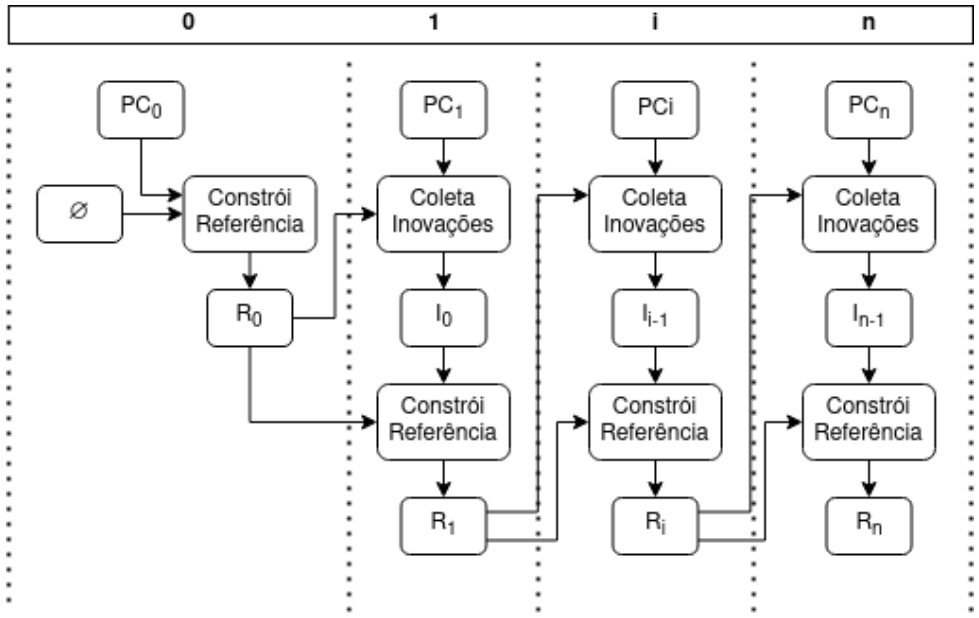

Figura 3.6: Fluxo de construções das nuvens de pontos de referência e de inovações a cada iteração.

Podemos definir uma nuvem de pontos como $PC = \{(x, y, z, r) \in \mathbb{R}^4\}$ sendo (x, y, z) as coordenadas geométricas e r a reflectância. Como neste trabalho o foco é apenas nos pontos geométricos, iremos abstrair o r, portanto, redefinimos $PC = \{(x, y, z) \in \mathbb{R}^3\}$. Para definições posteriores, denotaremos o conjunto de todas as nuvens de pontos de um determinado experimento do *Kitti* por PCT.

Seja n a quantidade total de quadros adquiridos dinamicamente com um LiDAR em um experimento do *Kitti*, e seja $PC_i \in PCT$, onde i representa uma nuvem de pontos em um determinado instante de tempo, e $PC_{i+1} \in PCT$ a nuvem de pontos no instante seguinte. Também vamos definir outro conjunto, que representa os pontos que possuem intersecção com PC_i e PC_{i+1}, seja $B = \{(x, y, z) \in \mathbb{R}^3 | (x, y, z) \in PC_i \cap^* PC_{i+1}\}$.

O operador $\cap^*$ é usado aqui para representar um tipo especial de intersecção. Na intersecção padrão advinda da teoria de conjuntos é feita a comparação exata dos pontos, mas neste caso a intersecção é descrita como pontos próximos uns aos outros em uma distância de até 20 centímetros. A busca destes pontos é realizada com a função *radius-*

Search utilizando a *k-d Tree*, para maiores detalhes do funcionamento desta função ver Seção 2.6 e ler documentação da biblioteca $nanoflann$ [31].

Então definimos o conjunto que representa a nuvem de pontos que contém pontos de inovação a cada iteração i por $I_i = PC_{i+1} - B_i$. A cada iteração também teremos a construção de uma nuvem de referência denominada por R, sendo $R_i = I_{i-1} \cup R_{i-1}$, dado que na primeira iteração teremos $R_0 = PC_0 \cup \varnothing$.

3.6 Experimento

Os módulos foram desenvolvidos e executados em um sistema operacional Linux Ubuntu 20.04.2 LTS, de arquitetura x64. Todo o código desenvolvido neste trabalho pode ser encontrado como código aberto na plataforma *GitHub* [41]. O módulo descrito na Seção 3.4 pode ser encontrado nas pastas "src/iip/oxt2pose" e o módulo da Seção 3.5 no caminho "src/iip/pose2reg". Para alinhar as coordenadas foi utilizado *Python*, versão 3.7, e para encontrar os pontos de inovação e montar o mapa foi utilizado a linguagem *C++*, versão 9.3.0 do *GCC*.

O experimento *2011_09_26_drive_0001*, sincronizado e retificado, da categoria cidade do conjunto de dados *Kitti* [42] foi utilizado. No total, foram processados 108 quadros de nuvem de pontos, com em média 122,026.50 pontos geométricos.

Capítulo 4

Resultados e Discussões

Utilizando o módulo descrito na Seção 3.5, foram geradas 108 nuvens de pontos de referência e 107 de inovação. A quantidade de pontos geométricos destas nuvens e o tamanho de cada arquivo gerado estão presentes na Tabela 4.1 e Tabela 4.2, respectivamente.

Iteração	**Quadro**	**Pontos**	**Inovações**	**Referência**
0	0000000000.ply	121015	121015	121015
1	0000000001.ply	121151	6221	127236
2	0000000002.ply	121483	4836	132072
3	0000000003.ply	121604	4005	136077
4	0000000004.ply	121779	3019	139096
5	0000000005.ply	121948	2012	141108
6	0000000006.ply	121848	1950	143058
7	0000000007.ply	121997	1762	144820
8	0000000008.ply	122227	812	145632
9	0000000009.ply	122402	750	146382
10	0000000010.ply	122486	810	147192
11	0000000011.ply	122554	738	147930
12	0000000012.ply	122460	799	148729
13	0000000013.ply	122566	793	149522
14	0000000014.ply	122501	502	150024
15	0000000015.ply	122455	638	150662
16	0000000016.ply	122296	753	151415
17	0000000017.ply	122260	633	152048
18	0000000018.ply	122219	660	152708
19	0000000019.ply	121870	507	153215
20	0000000020.ply	121860	515	153730

21	0000000021.ply	121514	613	154343
22	0000000022.ply	121316	541	154884
23	0000000023.ply	121321	632	155516
24	0000000024.ply	121180	590	156106
25	0000000025.ply	121376	515	156621
26	0000000026.ply	121544	555	157176
27	0000000027.ply	121556	529	157705
28	0000000028.ply	121524	621	158326
29	0000000029.ply	121340	607	158933
30	0000000030.ply	121129	520	159453
31	0000000031.ply	121214	604	160057
32	0000000032.ply	121239	663	160720
33	0000000033.ply	121237	522	161242
34	0000000034.ply	121266	721	161963
35	0000000035.ply	121210	713	162676
36	0000000036.ply	121194	629	163305
37	0000000037.ply	121197	548	163853
38	0000000038.ply	121378	619	164472
39	0000000039.ply	121535	618	165090
40	0000000040.ply	121587	502	165592
41	0000000041.ply	121467	489	166081
42	0000000042.ply	121387	553	166634
43	0000000043.ply	121230	631	167265
44	0000000044.ply	121149	558	167823
45	0000000045.ply	121226	573	168396
46	0000000046.ply	121463	621	169017
47	0000000047.ply	121505	614	169631
48	0000000048.ply	121607	613	170244
49	0000000049.ply	121875	548	170792
50	0000000050.ply	121799	531	171323
51	0000000051.ply	121735	559	171882
52	0000000052.ply	121740	575	172457
53	0000000053.ply	121923	544	173001
54	0000000054.ply	121937	635	173636
55	0000000055.ply	122072	607	174243

56	0000000056.ply	122170	509	174752
57	0000000057.ply	122223	706	175458
58	0000000058.ply	122415	630	176088
59	0000000059.ply	122405	655	176743
60	0000000060.ply	122421	725	177468
61	0000000061.ply	122306	954	178422
62	0000000062.ply	122266	835	179257
63	0000000063.ply	122270	855	180112
64	0000000064.ply	122376	842	180954
65	0000000065.ply	122474	850	181804
66	0000000066.ply	122627	835	182639
67	0000000067.ply	122763	923	183562
68	0000000068.ply	122850	722	184284
69	0000000069.ply	122945	679	184963
70	0000000070.ply	122973	751	185714
71	0000000071.ply	123038	589	186303
72	0000000072.ply	123007	736	187039
73	0000000073.ply	123012	560	187599
74	0000000074.ply	123048	527	188126
75	0000000075.ply	123092	410	188536
76	0000000076.ply	122965	482	189018
77	0000000077.ply	123029	456	189474
78	0000000078.ply	123006	573	190047
79	0000000079.ply	123026	491	190538
80	0000000080.ply	123176	411	190949
81	0000000081.ply	123136	376	191325
82	0000000082.ply	123152	483	191808
83	0000000083.ply	123109	504	192312
84	0000000084.ply	123053	404	192716
85	0000000085.ply	123124	364	193080
86	0000000086.ply	123033	422	193502
87	0000000087.ply	122955	273	193775
88	0000000088.ply	122978	299	194074
89	0000000089.ply	122919	440	194514
90	0000000090.ply	122987	503	195017

91	0000000091.ply	123040	430	195447
92	0000000092.ply	123020	421	195868
93	0000000093.ply	123127	450	196318
94	0000000094.ply	123140	411	196729
95	0000000095.ply	123162	374	197103
96	0000000096.ply	123253	440	197543
97	0000000097.ply	123259	484	198027
98	0000000098.ply	123174	466	198493
99	0000000099.ply	123181	341	198834
100	0000000100.ply	123118	441	199275
101	0000000101.ply	123014	401	199676
102	0000000102.ply	122964	402	200078
103	0000000103.ply	122876	481	200559
104	0000000104.ply	122718	497	201056
105	0000000105.ply	122543	469	201525
106	0000000106.ply	122669	414	201939
107	0000000107.ply	98322	410	202349

Tabela 4.1: Quantidade de pontos geométricos a cada nuvem de pontos dinamicamente adquirida, suas inovações e os pontos de cada referência por iteração.

Iteração	**Quadro**	**Pontos (MB)**	**Inovações (MB)**	**Referência (MB)**
0	0000000000.ply	2.40	2.40	2.40
1	0000000001.ply	3.00	0.16	2.50
2	0000000002.ply	3.00	0.12	2.70
3	0000000003.ply	3.00	0.10	2.80
4	0000000004.ply	3.00	0.08	2.80
5	0000000005.ply	3.00	0.05	2.90
6	0000000006.ply	3.00	0.05	2.90
7	0000000007.ply	3.00	0.05	3.00
8	0000000008.ply	3.00	0.02	3.00
9	0000000009.ply	3.00	0.02	3.00
10	0000000010.ply	3.00	0.02	3.00
11	0000000011.ply	3.00	0.02	3.00
12	0000000012.ply	3.00	0.02	3.10

13	0000000013.ply	3.00	0.02	3.10
14	0000000014.ply	3.00	0.01	3.10
15	0000000015.ply	3.00	0.02	3.10
16	0000000016.ply	3.00	0.02	3.10
17	0000000017.ply	3.00	0.02	3.10
18	0000000018.ply	3.00	0.02	3.20
19	0000000019.ply	3.00	0.01	3.20
20	0000000020.ply	3.00	0.01	3.20
21	0000000021.ply	3.00	0.02	3.20
22	0000000022.ply	3.00	0.01	3.20
23	0000000023.ply	3.00	0.02	3.20
24	0000000024.ply	3.00	0.02	3.20
25	0000000025.ply	3.00	0.01	3.30
26	0000000026.ply	3.00	0.01	3.30
27	0000000027.ply	3.00	0.01	3.30
28	0000000028.ply	3.00	0.02	3.30
29	0000000029.ply	3.00	0.02	3.30
30	0000000030.ply	3.00	0.01	3.30
31	0000000031.ply	3.00	0.02	3.30
32	0000000032.ply	3.00	0.02	3.40
33	0000000033.ply	3.00	0.01	3.40
34	0000000034.ply	3.00	0.02	3.40
35	0000000035.ply	3.00	0.02	3.40
36	0000000036.ply	3.00	0.02	3.40
37	0000000037.ply	3.00	0.01	3.40
38	0000000038.ply	3.00	0.02	3.50
39	0000000039.ply	3.00	0.02	3.50
40	0000000040.ply	3.00	0.01	3.50
41	0000000041.ply	3.00	0.01	3.50
42	0000000042.ply	3.00	0.01	3.50
43	0000000043.ply	3.00	0.02	3.50
44	0000000044.ply	3.00	0.02	3.50
45	0000000045.ply	3.00	0.02	3.50
46	0000000046.ply	3.00	0.02	3.60
47	0000000047.ply	3.00	0.02	3.60

48	0000000048.ply	3.00	0.02	3.60
49	0000000049.ply	3.00	0.01	3.60
50	0000000050.ply	3.00	0.01	3.60
51	0000000051.ply	3.00	0.02	3.60
52	0000000052.ply	3.10	0.02	3.60
53	0000000053.ply	3.10	0.01	3.70
54	0000000054.ply	3.10	0.02	3.70
55	0000000055.ply	3.10	0.02	3.70
56	0000000056.ply	3.10	0.01	3.70
57	0000000057.ply	3.10	0.02	3.70
58	0000000058.ply	3.10	0.02	3.70
59	0000000059.ply	3.10	0.02	3.70
60	0000000060.ply	3.10	0.02	3.80
61	0000000061.ply	3.10	0.02	3.80
62	0000000062.ply	3.10	0.02	3.80
63	0000000063.ply	3.10	0.02	3.80
64	0000000064.ply	3.10	0.02	3.90
65	0000000065.ply	3.10	0.02	3.90
66	0000000066.ply	3.10	0.02	3.90
67	0000000067.ply	3.10	0.02	3.90
68	0000000068.ply	3.10	0.02	3.90
69	0000000069.ply	3.10	0.02	3.90
70	0000000070.ply	3.10	0.02	4.00
71	0000000071.ply	3.10	0.02	4.00
72	0000000072.ply	3.10	0.02	4.00
73	0000000073.ply	3.10	0.01	4.00
74	0000000074.ply	3.10	0.01	4.00
75	0000000075.ply	3.10	0.01	4.00
76	0000000076.ply	3.10	0.01	4.00
77	0000000077.ply	3.10	0.01	4.10
78	0000000078.ply	3.10	0.02	4.10
79	0000000079.ply	3.10	0.01	4.10
80	0000000080.ply	3.10	0.01	4.10
81	0000000081.ply	3.10	0.01	4.10
82	0000000082.ply	3.10	0.01	4.10

83	0000000083.ply	3.10	0.01	4.10
84	0000000084.ply	3.10	0.01	4.10
85	0000000085.ply	3.10	0.01	4.10
86	0000000086.ply	3.10	0.01	4.20
87	0000000087.ply	3.10	0.01	4.20
88	0000000088.ply	3.10	0.01	4.20
89	0000000089.ply	3.10	0.01	4.20
90	0000000090.ply	3.10	0.01	4.20
91	0000000091.ply	3.10	0.01	4.20
92	0000000092.ply	3.10	0.01	4.20
93	0000000093.ply	3.10	0.01	4.20
94	0000000094.ply	3.10	0.01	4.20
95	0000000095.ply	3.10	0.01	4.20
96	0000000096.ply	3.10	0.01	4.20
97	0000000097.ply	3.10	0.01	4.30
98	0000000098.ply	3.10	0.01	4.30
99	0000000099.ply	3.10	0.01	4.30
100	0000000100.ply	3.10	0.01	4.30
101	0000000101.ply	3.10	0.01	4.30
102	0000000102.ply	3.10	0.01	4.30
103	0000000103.ply	3.10	0.01	4.30
104	0000000104.ply	3.10	0.01	4.30
105	0000000105.ply	3.10	0.01	4.30
106	0000000106.ply	3.10	0.01	4.40
107	0000000107.ply	2.50	0.01	4.40

Tabela 4.2: Tamanho dos arquivos de nuvens de pontos a cada iteração.

Foi também gerada a unificação do mapa pelo método *Iterative Closest Point (ICP) Point-to-Plane (Po2Pl)* como descrito na Equação 2.12 destes 108 quadros, e os resultados são apresentados abaixo, na Tabela 4.3.

Pontos	**Tamanho (MB)**
1341608.00	31

Tabela 4.3: Quantidade de pontos e tamanho do arquivo gerado a partir da unificação de pontos por *ICP*.

Podemos então observar a partir das tabelas acima a enorme redução dos pontos geométricos gerados pelo método proposto em comparação com o *Iterative Closest Point (ICP)*. Essa comparação é mostrada na Tabela 4.4 e na Tabela 4.5, e também pode ser estabelecida visualmente através da Figura 4.1. O método proposto neste estudo é denominado *Iterative Innovation Point (IIP)* e está na tabela abaixo (Tabela 4.4).

Método	**Pontos**	**Tamanho (MB)**
IIP	202349.00	4.40
ICP	1341608.00	31

Tabela 4.4: Método, quantidade de pontos geométricos e tamanho do arquivo final.

Redução de Pontos (%)	**Redução de Tamanho (%)**
84.92	85.81

Tabela 4.5: Porcentagem da redução de pontos geométricos e tamanho do arquivos dos métodos da Tabela 4.4

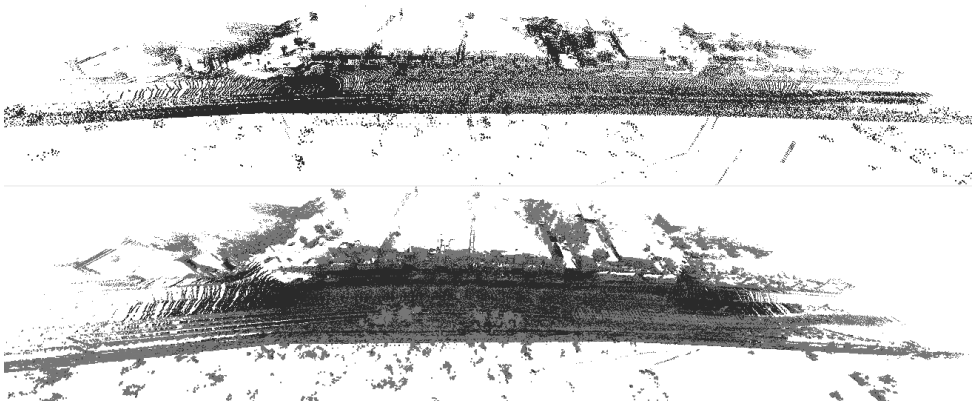

Figura 4.1: Unificação do mapa utilizando o método proposto neste trabalho, primeiro mapa, e mapa gerado pelo *ICP*, segundo mapa.

Podemos também notar um grande ganho de redução por nuvem de ponto a cada iteração, caso seja preciso transmitir essas nuvens dinamicamente adquiridas, teríamos uma redução de 99.38 % em média com desvio padrão de 0.68 %. Um exemplo da nuvem de pontos gerada a cada iteração pode ser visualizado na Figura 4.2.

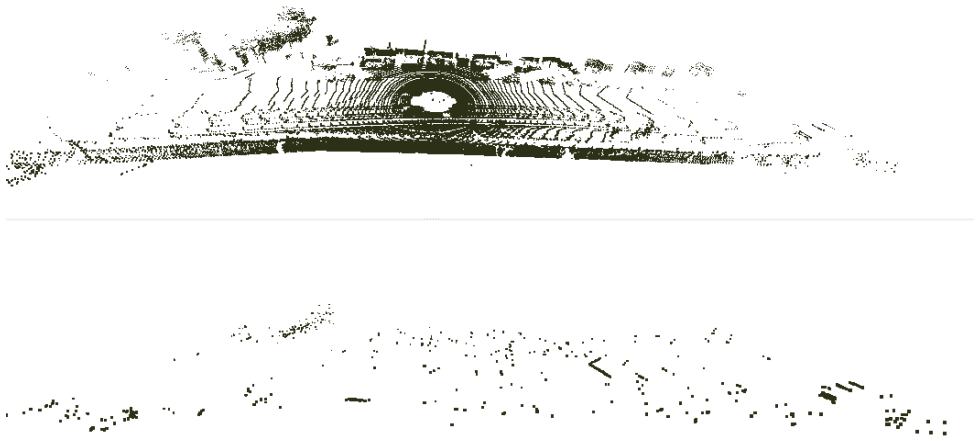

Figura 4.2: Nuvem de pontos na iteração 10, imagem acima, e a nuvem de pontos com pontos de inovações, imagem abaixo.

Capítulo 5

Conclusão

Foi proposto neste trabalho um método de redução de redundância geométrica, também denominado de redundância temporal ou inter-quadros, de nuvens de pontos adquiridas dinamicamente por um sensor *Light Detection And Ranging (LiDAR)*, sendo este sensor comumente utilizado em sistemas automotivos e autônomos. Este método propôs o uso de busca e remoção de pontos muito próximos uns aos outros através de uma busca por raio em uma *k-D Tree*.

Antes dessa identificação dos pontos com muita proximidade, situação que não traria muita informação nova entre um quadro num instante de tempo e o quadro subsequente, as nuvens de pontos tiveram que ser transformadas para que os pontos se encontrassem no mesmo sistema de coordenadas e em coordenadas globais.

Os resultados mostraram que este método apresentou 84.92 % de redução de pontos geométricos em comparação com o método *Iterative Closest Point (ICP)* e 85.81 % de redução no tamanho do arquivo com o mapa final. Também foi possível identificar que a cada iteração, uma redução de 99.38 % era proporcionada em média com desvio padrão de 0.68 % dos pontos geométricos.

Referências

[1] Gonzalez, Rafael C, Richard E Woods *et al.*: *Digital image processing*, 2002. ix, 7, 8

[2] Rusu, Radu Bogdan e Steve Cousins: *3d is here: Point cloud library (pcl)*. Em *2011 IEEE international conference on robotics and automation*, páginas 1–4. IEEE, 2011. ix, 10

[3] Singu, Poojitha, VS Ramya Lakshmi e NR Raajan: *Point cloud human posture estimation using single rgb image*. Materials Today: Proceedings, 2020. ix, 10

[4] Tier IV, Inc.: *Autoware tools*. `https://tools.tier4.jp/`, acesso em: 20/05/2021. ix, 10

[5] Hua, Binh Son, Minh Khoi Tran e Sai Kit Yeung: *Pointwise convolutional neural networks*. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 984–993, 2018. ix, 10

[6] Watkinson, John: *The MPEG Handbook: MPEG-1, MPEG-2, MPEG-4*. Taylor & Francis, 2004. ix, 13

[7] Spong, Mark W, Seth Hutchinson, Mathukumalli Vidyasagar *et al.*: *Robot modeling and control*, volume 3. wiley New York, 2006. ix, 14, 16, 17

[8] Geiger, Andreas, Philip Lenz, Christoph Stiller e Raquel Urtasun: *Vision meets robotics: The kitti dataset*. International Journal of Robotics Research (IJRR), 2013. ix, 6, 12, 21, 22

[9] Preparata, Franco P e Michael I Shamos: *Computational geometry: an introduction*. Springer Science & Business Media, 2012. x, 11

[10] Besl, Paul J e Neil D McKay: *Method for registration of 3-d shapes*. Em *Sensor fusion IV: control paradigms and data structures*, volume 1611, páginas 586–606. International Society for Optics and Photonics, 1992. xi, 18

[11] Chen, Yang e Gérard Medioni: *Object modelling by registration of multiple range images*. Image and vision computing, 10(3):145–155, 1992. xi, 18

[12] Heineke, Kersten, Philipp Kampshoff e Timo Möller: *From no mobility to future mobility: Where COVID-19 has accelerated change*. (December 2020):1–17, 2020. 1

[13] Holland-Letz, Daniel, Benedikt Kloss, Matthias Kässer e Thibaut Müller: *Start me up: Where mobility investments are going*. (March):8, 2019. 1

[14] Twinn, Ian, Navaid Qureshi, Maria López Conde, Carlos Garzón Guinea, Daniel Perea Rojas, Jiayuan Luo e Harsh Gupta: *The Impact of COVID-19 on Logistics*. página 5, 2020. https://www.ifc.org/wps/wcm/connect/2d6ec419-41df-46c9-8b7b-96384cd36ab3/IFC-Covid19-Logistics-final_web.pdf?MOD=AJPERES&CVID=naqOED5. 2

[15] Hörl, Sebastian; Ciari, Francesco; Axhausen, Kay W.: *Recent perspectives on the impact of autonomous vehicles*. https://doi.org/10.3929/ethz-a-010025751, 2009. 2

[16] Bimbraw, Keshav: *Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology*. Em *2015 12th international conference on informatics in control, automation and robotics (ICINCO)*, volume 1, páginas 191–198. IEEE, 2015. 2

[17] Geyer, Jakob, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis e Peter Schuberth: *A2d2: Audi autonomous driving dataset*, 2020. 2, 21

[18] Dikmen, Murat e Catherine M. Burns: *Autonomous driving in the real world: Experiences with tesla autopilot and summon*. Em *Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, Automotive'UI 16, página 225–228, New York, NY, USA, 2016. Association for Computing Machinery, ISBN 9781450345330. https://doi.org/10.1145/3003715.3005465. 2

[19] committee, On Road Automated Driving (ORAD): *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, apr 2021. https://doi.org/10.4271/J3016_202104. 2

[20] Organization, World Health *et al.*: *Global status report on road safety 2018: Summary*. Relatório Técnico, World Health Organization, 2018. 3

[21] Marchant, Gary E e Rachel A Lindor: *The coming collision between autonomous vehicles and the liability system*. Santa Clara L. Rev., 52:1321, 2012. 3

[22] Google: *Google maps*. https://www.google.com.br/maps, acesso em: 24/05/2021. 3

[23] Mobile, Waze: *Waze*. https://www.waze.com, acesso em: 24/05/2021. 3

[24] Hussain, Rasheed e Sherali Zeadally: *Autonomous cars: Research results, issues, and future challenges*. IEEE Communications Surveys & Tutorials, 21(2):1275–1313, 2018. 5

[25] Toriwaki, Junichiro e Hiroyuki Yoshida: *Fundamentals of three-dimensional digital image processing*. Springer Science & Business Media, 2009. 8

[26] Knuth, Donald Ervin: *The art of computer programming*, volume 3. Pearson Education, 1997. 11

[27] Zhang, Zhengyou: *Iterative point matching for registration of free-form curves and surfaces.* International Journal of Computer Vision, 13(2):119–152, 1994, ISSN 09205691. 11

[28] Velodyne Lidar, Inc.: *Velodyne lidar.* `https://velodynelidar.com/`, acesso em: 28/04/2021. 12

[29] Velodyne Lidar, Inc.: *Velodyne hdl-64e.* `https://velodynelidar.com/products/hdl-64e/`, acesso em: 28/04/2021. 12

[30] Qfix: *RT GNSS-aided inertial measurement systems.* 3304(January):1–148, 2012. 13

[31] Blanco, Jose Luis e Pranjal Kumar Rai: *nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees.* `https://github.com/jlblancoc/nanoflann`, 2014. 14, 27

[32] LaValle, Steven M: *Planning algorithms.* Cambridge university press, 2006. 16

[33] Petrovski, Ivan G e Toshiaki Tsujii: *Digital satellite navigation and geophysics: A practical guide with GNSS signal simulator and receiver laboratory.* Cambridge University Press, 2012. 17

[34] Souto, André e Ricardo Queiroz: *Transformada RAHT com Predição Inter-Quadros para Compressão de Nuvem de Pontos.* páginas 22–25, 2020. 18

[35] Garcia, Diogo C e Ricardo L de Queiroz: *Intra-frame context-based octree coding for point-cloud geometry.* Em *2018 25th IEEE International Conference on Image Processing (ICIP)*, páginas 1807–1811. IEEE, 2018. 19

[36] Kammerl, Julius, Nico Blodow, Radu Bogdan Rusu, Suat Gedikli, Michael Beetz e Eckehard Steinbach: *Real-time compression of point cloud streams.* Em *2012 IEEE International Conference on Robotics and Automation*, páginas 778–785. IEEE, 2012. 19

[37] Li, Yumin, Yanmin Wang *et al.*: *An accurate registration method based on point clouds and redundancy elimination of lidar data.* The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 37(B5):605–610, 2008. 19

[38] Schauer, Johannes e Andreas Nüchter: *The peopleremover—removing dynamic objects from 3-d point cloud data by traversing a voxel occupancy grid.* IEEE robotics and automation letters, 3(3):1679–1686, 2018. 19

[39] Du, Ruoyu e Hyo Jong Lee: *A novel compression algorithm for lidar data.* Em *2012 5th International Congress on Image and Signal Processing*, páginas 987–991. IEEE, 2012. 19

[40] Wang, Peng, Xinyu Huang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng e Ruigang Yang: *The apolloscape open dataset for autonomous driving and its application.* IEEE transactions on pattern analysis and machine intelligence, 2019. 21

[41] Cunha, Dayanne Fernandes da: *Iip, iterative innovation points.* `https://github.com/Dayof/point-cloud-lib`, acesso em: 12/05/21. 27

[42] Geiger, Andreas, Philip Lenz, Christoph Stiller e Raquel Urtasun: *kittiweb.* `http://www.cvlibs.net/datasets/kitti/raw_data.php`, acesso em: 10/05/21. 27