UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

CENTRO TECNOLÓGICO

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

EDUARDO MAX AMARO AMARAL

Detecção e Rastreamento de Veículos em Movimento para Automóveis Robóticos Autônomos

VITÓRIA

2015

EDUARDO MAX AMARO AMARAL

Detecção e Rastreamento de Veículos em Movimento para Automóveis Robóticos Autônomos

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Informática.

VITÓRIA

2015

EDUARDO MAX AMARO AMARAL

Detecção e Rastreamento de Veículos em Movimento para Automóveis Robóticos Autônomos

COMISSÃO EXAMINADORA

Prof. Dr. Alberto Ferreira de Souza
Universidade Federal do Espírito Santo
Orientador

Profª. Drª. Claudine Badue
Universidade Federal do Espírito Santo
Coorientadora

Prof. Dr. Thiago Oliveira dos Santos
Universidade Federal do Espírito Santo

Profª. Drª. Karin Satie Komati
Instituto Federal do Espírito Santo

Vitória, 26 de maio de 2015.

AGRADECIMENTOS

Foi uma experiência enriquecedora e de plena superação escrever uma dissertação de Mestrado. A cada tentativa de buscar respostas às aflições, um novo desafio aparecia trazendo consigo novas aflições carregadas de visões e ideias mais amadurecidas e com diferentes perspectivas.

Para aqueles que compartilharam comigo desse momento, eu gostaria de agradecer:

Ao meu orientador, professor Dr. Alberto Ferreira de Souza por me dar importantes e significativos conselhos.

À minha coorientadora, professora Drª. Claudine Badue, pela dedicação, orientação e revisão técnica deste projeto e pelos valiosos ensinamentos que tornaram este trabalho possível.

Ao professor Dr. Thiago Oliveira dos Santos pelos relevantes ensinamentos técnicos que foram fundamentais para o desenvolvimento deste trabalho.

Aos colegas do LCAD pelo mútuo aprendizado durante nossa convivência e pela confiança em mim depositada. Em especial, agradeço ao Filipe Wall Mutz por estar sempre de prontidão para debater ideias e responder dúvidas.

Aos meus pais, Elesonte e Maximina, pela paciência e compreensão que tiveram comigo em todos os momentos desta e de outras caminhadas, sempre me acolhendo nos momentos de aflição e me apoiando em meus projetos.

À minha esposa Kênia e ao meu filho Henrique, sou grato pela compreensão da minha ausência e da falta de tempo para os papéis integrais de marido e de pai, além de todo suporte e apoio sem o qual seria muito mais difícil alcançar qualquer objetivo.

Por fim, a todos aqueles que, direta ou indiretamente, colaboraram no desenvolvimento deste trabalho.

RESUMO

Neste trabalho, foi investigado o problema de detecção e rastreamento de objetos em movimento (*detection and tracking of moving objects* - DATMO) para veículos robóticos autônomos. DATMO envolve a detecção de cada objeto em movimento no ambiente ao redor do veículo autônomo e o seu rastreamento, i.e., a estimativa do seu estado (e.g., posição, orientação e velocidade) ao longo do tempo. O veículo autônomo precisa estimar os estados dos objetos ao longo do tempo, de forma que possa predizê-los alguns segundos mais tarde para fins de mapeamento, localização e navegação.

Foi proposto um sistema de DATMO para detecção e rastreamento de múltiplos veículos em movimento no ambiente ao redor de um veículo autônomo usando um sensor *Light Detection and Ranging* (LIDAR) 3D. O sistema de DATMO proposto opera em três etapas: segmentação, associação e rastreamento. A cada varredura do sensor, na etapa de segmentação, os pontos 3D associados ao plano do solo são removidos; a nuvem de pontos 3D é segmentada em agrupamentos de pontos usando a distância Euclidiana, sendo que cada agrupamento representa um objeto no ambiente; e os agrupamentos relacionados aos meios-fios são removidos. Na etapa de associação, os objetos observados na varredura atual do sensor são associados aos mesmos objetos observados em varreduras anteriores usando o algoritmo do vizinho mais próximo (*nearest neighbor*). Finalmente, na etapa de rastreamento, os estados dos objetos são estimados usando um filtro de partículas. Os objetos com velocidade acima de um determinado limiar são considerados veículos em movimento.

O desempenho do sistema de DATMO proposto foi avaliado usando dados de um sensor LIDAR 3D, além de dados de outros sensores, coletados por um veículo autônomo ao longo de uma volta pelo anel viário do campus da Universidade Federal do Espírito Santo (UFES). Os resultados experimentais mostraram que o sistema de DATMO proposto foi capaz de detectar e rastrear com bom desempenho múltiplos veículos em movimento no ambiente ao redor do veículo autônomo.

ABSTRACT

In this work, it was investigated the problem of detection and tracking of moving objects (DATMO) for autonomous robotic vehicles. DATMO involves the detection of each moving object in the environment around the autonomous vehicle and its tracking, i.e., estimation of its state (e.g., position, orientation and velocity) over time. The autonomous vehicle needs to estimate the state of objects over time, so that it can predict their states a few seconds later for purposes of mapping, localization and navigation.

It was proposed a DATMO system for the detection and tracking of multiple moving vehicles in the environment around an autonomous vehicle using a Light Detection and Ranging sensor (LIDAR) 3D. The proposed DATMO system operates in three steps: segmentation, association and tracking. At each sensor scan, in the segmentation step, the 3D points associated with the ground plane are removed; the 3D point cloud is segmented into clusters of points using the Euclidean distance, wherein each cluster represents an object in the environment; and the clusters related to curbs are removed. In association step, the objects observed in the current scan sensor are associated with the same objects observed in previous scans using the nearest neighbor algorithm. Finally, in the tracking step, the states of objects are estimated using a particle filter. Objects with velocity above a given threshold are considered moving vehicles.

The performance of the proposed DATMO system was evaluated using data from a LIDAR 3D sensor, besides data from other sensors, collected by an autonomous vehicle along a ring road around the campus of the Federal University of Espírito Santo (Universidade Federal do Espírito Santo - UFES). The experimental results showed that the proposed DATMO system was able to detect and track with good performance multiple moving vehicles on the environment around the autonomous vehicle.

LISTA DE FIGURAS

LISTA DE TABELAS

Sumário

1 INTRODUÇÃO

Veículos robóticos autônomos podem trazer uma série de benefícios para a sociedade, incluindo o uso mais eficiente de combustíveis, conforto e conveniência, auxílio de pessoas com necessidades especiais, como as com dificuldades de locomoção, e principalmente a prevenção de acidentes rodoviários.

Os acidentes de transporte são uma das principais causas de morte no Brasil e no mundo. Acidentes de transporte incorporam, além dos comumente denominados “acidentes de trânsito”, outros acidentes derivados das atividades de transporte, como aéreo e por água [1]. As taxas de óbito em acidentes de transporte do Brasil são extremamente elevadas e bem acima da média internacional [1] [2]. Com sua taxa de 23 mortes em acidentes de transporte por 100 mil habitantes, o Brasil ocupa a quarta posição entre os 101 países elencados [1].

Devido a isso, justificam-se os estudos sobre sistemas que possam informar com antecedência a existência de riscos de colisões com outros veículos e auxiliar o motorista na dirigibilidade. Muitos trabalhos já foram desenvolvidos neste sentido, como os sistemas de assistência ao condutor na detecção de sonolência do motorista, mudança de faixa, frenagem de emergência, entre outros [3] [4] [5] [6]. Sensoriamento e controle automatizado constituem importantes tecnologias que, se aplicadas aos futuros veículos, poderão salvar milhares de vidas que seriam perdidas em acidentes de trânsito [7].

Atualmente, diversas empresas estão direcionando esforços para o desenvolvimento de veículos autônomos. Google desenvolveu um veículo autônomo baseado no Toyota Prius e já percorreu mais de 500.000 km autonomamente [8]. Em maio de 2014, a empresa apresentou uma versão inicial do protótipo de um veículo sem volante, pedal de acelerador ou pedal de freio, e afirmou que os softwares e os sensores fazem todo o trabalho [9]. Outro veículo autônomo foi desenvolvido pelo AutoNOMOS Labs [10]. Este veículo é baseado no Volkswagen Passat e está funcionando com uma permissão especial para navegar pelas ruas de Berlim [11]. Volvo apresentou

um veículo capaz de dirigir autonomamente e, para isso, o carro está equipado com um sistema para detectar animais, pedestres, faixas de rodovias e barreiras [12]. Nissan e NASA se unirão para desenvolver veículos autônomos e pretendem apresentar um carro com a capacidade de navegar autonomamente nas ruas das cidades até 2020 [13].

1.1 Tarefas Fundamentais em Robótica Autônoma

Mapeamento e localização simultâneos (*simultaneous localization and mapping* – SLAM), detecção e rastreamento de objetos em movimento (*detection and tracking of moving objects* – DATMO) e planejamento de movimento são tarefas fundamentais na robótica autônoma. SLAM envolve a construção de um mapa do ambiente (lista de objetos no ambiente e suas posições) ao redor do veículo autônomo à medida que o veículo está navegando no ambiente e, simultaneamente, a estimativa do estado (e.g., posição e orientação) do veículo em relação ao mapa [14]. O veículo autônomo precisa saber como é o ambiente ao seu redor e onde ele está no ambiente para fins de navegação. Para construir um mapa, o veículo autônomo requer a habilidade de detectar objetos estáticos, de forma que possa representá-los no mapa, bem como a capacidade de detectar e rastrear objetos em movimento, de forma que possa filtrá-los do mapa.

DATMO envolve a detecção de cada objeto em movimento no ambiente ao redor do veículo autônomo e o seu rastreamento, i.e., estimativa do seu estado (e.g., posição, orientação e velocidade) ao longo do tempo [15]. O veículo autônomo precisa estimar os estados dos objetos ao longo do tempo, de forma que possa predizê-los alguns segundos mais tarde para fins de SLAM e planejamento de movimento.

Planejamento de movimento envolve o planejamento de uma trajetória (lista de comandos e respectivos tempos de execução) a partir da posição inicial do robô até uma localização específica no ambiente, que mantém uma distância segura de obstáculos (objetos estáticos e móveis) e respeita restrições do contexto em que atua,

tais como permanecer na pista correta e manter uma velocidade segura [16]. O robô precisa executar os comandos corretos de forma a atingir seu objetivo, que pode estar relacionado a chegar a um lugar específico, enquanto evita colisões com obstáculos e obedece a limitações do meio em que o robô opera. Para navegar, o robô requer a habilidade de construir um mapa e localizar-se em relação a este mapa – SLAM, bem como a capacidade de detectar e rastrear objetos móveis – DATMO, de forma a evitar colisões com objetos estáticos e móveis.

SLAM e planejamento de movimento não serão tratados neste trabalho, dado que sistemas eficientes de mapeamento e localização, bem como planejamento de movimento, já foram desenvolvidos pelo Laboratório de Computação de Alto Desempenho (LCAD) do Departamento de Informática (DI) da Universidade Federal do Espírito Santo (UFES), ao qual este trabalho está associado.

1.2 *Detection and Tracking of Moving Objects* (DATMO)

No contexto de DATMO, um veículo autônomo navega em um ambiente dinâmico e os sensores são montados no próprio veículo, que pode se mover em alta velocidade. O veículo autônomo precisa perceber o ambiente ao seu redor com base em seus sensores para detectar e rastrear todos os objetos que se movem em sua vizinhança. Para uma tomada de decisão em alto nível, o veículo autônomo precisa estimar o estado de cada objeto em movimento com base nas medições do sensor. Para planejamento de trajetória em baixo nível, o veículo autônomo precisa estimar o espaço livre para a navegação. As estimativas do estado dos objetos em movimento precisam ser produzidas a uma taxa elevada o suficiente para serem úteis no ciclo percepção-planejamento-controle. Falsos negativos (i.e., não detectar um objeto presente) tendem a ser muito perigosos, ao passo que falsos positivos (i.e., detectar um objeto que não está presente) são mais aceitáveis [17].

Diversas abordagens foram propostas para a solução do problema de DATMO [17] [18] [19] [20] [21] [22]. As abordagens para DATMO eram quase exclusivamente im-

plementadas com base nos dados adquiridos por sensores de alcance 2D [18] [20] [21] [22] [23], principalmente sensores laser, chamados de *Light Detection and Ranging* (LIDAR) 2D. Estes sensores digitalizam o ambiente ao longo de um plano dentro de um ângulo de visão limitado e, assim, os objetos acima ou abaixo deste plano não podem ser detectados.

Recentemente, sensores LIDAR 3D foram introduzidos comercialmente. Embora exista uma enorme quantidade de abordagens propostas para DATMO usando sensores LIDAR 2D, o problema de DATMO usando sensores LIDAR 3D ainda é pouco abordado e investigado [15] [24] [25]. Uma das principais razões é a grande quantidade de dados fornecidos por sensores LIDAR 3D. A quantidade de dados de um sensor LIDAR 3D é geralmente muitas vezes maior do que a de um LIDAR 2D. Por conseguinte, a manipulação dos dados de um sensor LIDAR 3D requer estruturas de dados e algoritmos eficientes. Além disso, a extração e a interpretação de informação de dados da geometria em 3D são muito mais complexas do que a em 2D [26]. Contudo, apesar de encontrarem soluções satisfatórias para o problema, as abordagens para DATMO que usam sensores LIDAR 2D não são capazes de criar modelos do ambiente tão ricos como aquelas que usam sensores LIDAR 3D.

Neste trabalho, foi investigado o problema de DATMO para veículos autônomos. Foram estudadas várias abordagens para a solução deste problema e foi proposto um sistema de DATMO para detecção e rastreamento de múltiplos veículos em movimento no ambiente ao redor de um veículo autônomo usando um sensor LIDAR 3D.

Existem vários desafios a serem superados na tarefa de detecção e rastreamento de veículos em movimento, pois veículos possuem uma enorme gama de características, tais como formas, tamanhos e cores. Além disso, veículos em movimento devem ser detectados em cenários ao ar livre, ou seja, devem ser identificados no âmbito de um fundo túrbido e em cenas altamente dinâmicas, uma vez que veículos e sensores estão em movimento. Igualmente desafiador, veículos em movimento aparecem em diferentes ângulos de visão.

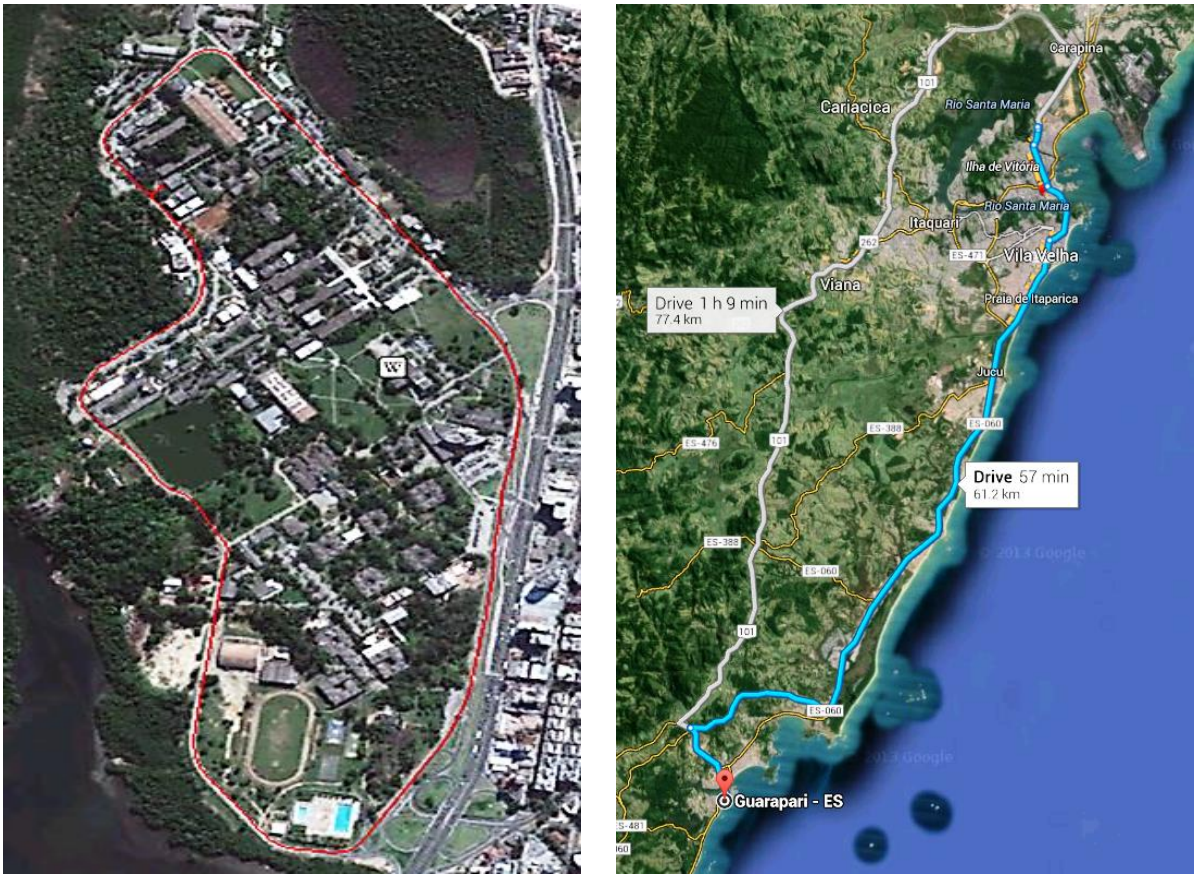
O sistema de DATMO proposto opera em três etapas: segmentação, associação e rastreamento. A cada varredura do sensor LIDAR 3D, após a conversão dos dados do sensor em uma nuvem de pontos 3D com coordenadas globais, na etapa de segmentação, os pontos associados ao plano do solo são removidos; a nuvem de pontos é segmentada em agrupamentos de pontos usando a distância Euclidiana, sendo que cada agrupamento representa um objeto no ambiente; e os agrupamentos relacionados aos meios-fios são removidos. Na etapa de associação, os objetos observados na varredura atual do sensor são associados aos mesmos objetos observados em varreduras anteriores usando o algoritmo do vizinho mais próximo (*nearest neighbor*). Finalmente, na etapa de rastreamento, os estados dos objetos são estimados usando um filtro de partículas. Os objetos com velocidade acima de um determinado limiar são considerados veículos em movimento.

1.3 Motivação

Este trabalho está inserido em uma linha de pesquisa do LCAD. Um dos objetivos dessa linha de pesquisa é a navegação autônoma do *Intelligent Autonomous Robot Automobile* (IARA) (Figura 1). IARA foi construído usando um automóvel de passeio *Ford Escape Hybrid* adaptado com sensores, mecanismos de controle e um conjunto de computadores instalados em seu porta-malas.

Figura 1: *Intelligent Autonomous Robotic Automobile* (IARA)

Como forma de validar a navegação autônoma, dois percursos foram definidos: a Volta da UFES e a Ida a Guarapari. Na Volta da UFES, o objetivo é desenvolver pesquisas que confiram ao IARA as habilidades necessárias para realizar uma volta ao redor do campus de Goiabeiras da UFES de forma autônoma. Este campus possui um anel viário com 3.570 m (Figura 2(a)). Na Ida a Guarapari, o objetivo é desenvolver pesquisas que viabilizem o aperfeiçoamento do IARA, de modo a torná-lo capaz de realizar uma viagem partindo do campus de Goiabeiras da UFES com destino à cidade de Guarapari de forma autônoma. O percurso da UFES a Guarapari possui 58,5 km (Figura 2(b)) e, neste caminho, todas as leis de trânsito terão que ser consideradas pelos algoritmos que controlarão IARA.



(a) (b)

Figura 2: (a) Caminho da Volta da UFES. (b) Caminho da Ida a Guarapari.

A principal motivação deste trabalho é contribuir para o alcance dos objetivos mencionados acima por meio do desenvolvimento de um sistema de DATMO capaz de

detectar e rastrear veículos em movimento no ambiente ao redor do IARA, com desempenho suficiente para permitir ao IARA uma navegação autônoma segura, tanto na Volta da UFES quanto na Ida a Guarapari.

1.4 Objetivos

O objetivo deste trabalho é desenvolver um sistema de DATMO para detecção e rastreamento de veículos em movimento no ambiente ao redor do IARA usando um sensor LIDAR 3D.

Para isso, será necessário: (i) estudar as funcionalidades essenciais a um veículo robótico autônomo; (ii) investigar os principais sistemas de DATMO usados por veículos autônomos e identificar as principais características destes sistemas que são úteis para a detecção e rastreamento de veículos em movimento usando um sensor LIDAR 3D; (iii) implementar um sistema de DATMO para detecção e rastreamento de veículos em movimento usando um sensor LIDAR 3D; e (iv) avaliar o desempenho do sistema de DATMO proposto usando dados de um sensor LIDAR 3D, além de dados de outros sensores, coletados por IARA ao longo de uma volta pelo anel viário do campus da UFES.

1.5 Contribuições

A principal contribuição deste trabalho foi o desenvolvimento de um sistema de DATMO para detecção e rastreamento de múltiplos veículos em movimento no ambiente ao redor de um veículo robótico autônomo usando um sensor LIDAR 3D. O sistema de DATMO proposto opera em três etapas: segmentação dos dados do sensor usando distância Euclidiana; associação dos dados usando o algoritmo do vizinho mais próximo; e rastreamento, i.e., estimativa do estado (posição, orientação e velocidade), dos veículos em movimento no ambiente usando um filtro de partículas. Até onde pudemos examinar na literatura, a combinação de técnicas que em-

pregamos para resolver o problema de DATMO é única. Além disso, a abordagem adotada no desenvolvimento deste trabalho usa um sensor LIDAR 3D de 32 feixes para percepção do ambiente ao redor do robô. Mesmo com um sistema de percepção muito mais reduzido do que o comumente utilizado (e.g., sensor LIDAR 3D de 64 feixes), o DATMO desenvolvido se mostrou capaz de detectar e rastrear veículos em movimento e desta forma contribui para transpor essa limitação.

Outra contribuição deste trabalho foi a avaliação do desempenho do sistema de DATMO proposto usando dados de um sensor LIDAR 3D, além de dados de outros sensores, coletados em diferentes datas por IARA ao longo de uma volta pelo anel viário do campus de Goiabeiras da UFES. Os resultados experimentais mostraram que o sistema de DATMO proposto foi capaz de detectar e rastrear com bom desempenho múltiplos veículos em movimento no ambiente ao redor de IARA.

1.6 Organização do Texto

O texto desta dissertação está organizado da seguinte forma.

Depois desta introdução, no Capítulo 2, é realizada uma revisão dos trabalhos relacionados a este trabalho. Neste capítulo, são apresentados os principais métodos de DATMO para veículos robóticos autônomos encontrados na literatura e são discutidas as vantagens e desvantagens de cada método.

No Capítulo 3, é apresentado o sistema de DATMO proposto neste trabalho para detecção e rastreamento de veículos em movimento no ambiente ao redor de um veículo autônomo.

No Capítulo 4, são descritos a metodologia experimental e os materiais usados para desenvolver e avaliar o desempenho do sistema de DATMO proposto. Neste capítulo, são descritos o *hardware* e o *software* usados para desenvolver e avaliar o sistema de DATMO proposto, os detalhes de implementação do sistema de DATMO desenvolvido, e a base de dados usada para avaliar o sistema de DATMO proposto.

No Capítulo 5, são demonstrados os resultados experimentais obtidos.

No Capítulo 6, são apresentadas as conclusões e possíveis direções para trabalhos futuros.

2 TRABALHOS RELACIONADOS

Em [15], os autores afirmam que a necessidade de DATMO nasceu com o primeiro interesse em veículos inteligentes e autônomos na década de 80. Um grande projeto exploratório foi lançado na Europa em 1987 sob o nome de PROMETHEUS - *Eureka PROMETHEUS Project (PROgraMme for a European Traffic of Highest Efficiency and Unprecedented Safety*, 1987-1995), seguido de uma série de iniciativas no Japão e Estados Unidos.

Uma equipe da *Stanford University* liderada por S. Thrun desenvolveu um veículo autônomo, denominado *Stanley*, que empregava câmeras para inferir a parte de interesse mais distante da estrada à sua frente [27]. *Stanley* foi capaz de se deslocar autonomamente por 142 milhas (~229Km) dentro do contexto de uma competição promovida em 2005 pela *Defense Advanced Research Projects Agency* (DARPA) dos USA denominada *DARPA Grand Challenge*. Em 2007, a DARPA promoveu uma *Grand Challenge* em percurso urbano (*DARPA Urban Challenge* - http://www.darpa.mil/grandchallenge). Nesta, os veículos autônomos precisavam respeitar regras de trânsito e coabitar com outros veículos não participantes da corrida. Seis veículos autônomos conseguiram completar o percurso. O primeiro lugar ficou com a *Carnegie Mellon University*, com o veículo chamado *Boss* [28] (Figura 3).

Figura 3: Boss: veículo vencedor do DARPA *Urban Challenge* desenvolvido pela equipe *Tartan Racing (Carnegie Mellon University)* [28]

O segundo lugar ficou com *Stanford* (*Junior*), que completou a prova em 4 horas [29]. Veja Figura 4.

Figura 4: Junior: veículo desenvolvido pela equipe da *Stanford University* para DARPA *Urban Challenge* [29]

Quase a totalidade dos veículos que participaram das *DARPA Grand Challenges* empregou câmeras e sensores laser (*Laser Range Scan* - LRS, ou *Light Detection And Ranging* - LIDAR) para examinar o terreno à frente e realizar, juntamente com sensores GPS, sua localização e mapeamento simultâneos (SLAM) e, a partir dos mapas, evitar obstáculos e perseguir os objetivos das provas.

Para aplicações de DATMO, os lasers 2D são tipicamente montados horizontalmente sobre o veículo autônomo e se caracterizam por produzir linhas de varredura paralelas ao chão. Recentemente os lasers evoluíram de 2D para lasers 3D. O princípio de operação é semelhante a um *scanner* 2D padrão, mas mais dados são recuperados, empregando mais feixes e mais receptores. O primeiro uso com sucesso deste dispositivo no campo de detecção de veículo foi apresentado no *DARPA Urban Challenge* [15] (Figura 5).

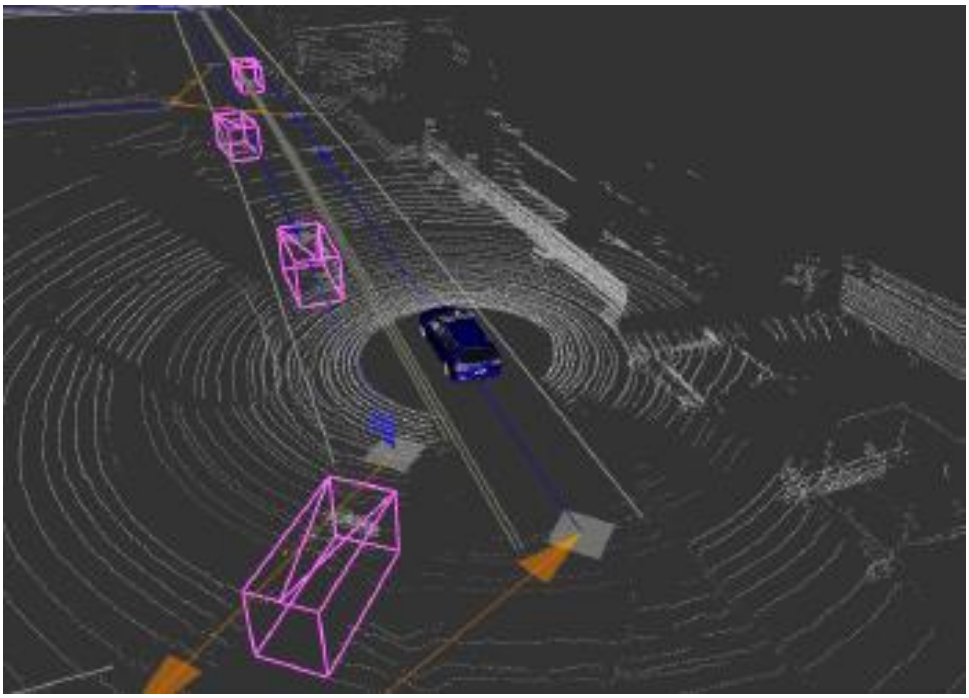


Figura 5: Exemplo de uma leitura 3D obtida de um LIDAR 3D. As caixas de cor rosa são veículos rastreados [15]

Seja qual for a tecnologia utilizada, a maioria dos algoritmos desenvolvidos para aplicações de rastreamento de objetos são baseados no uso de um clássico processo de três etapas: a detecção e segmentação de dados, uma etapa de associação, em seguida, um passo de estimativa e rastreamento [30] [31] [32] (Figura 6).

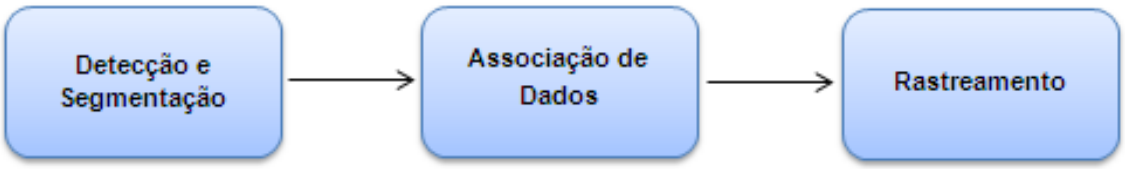


Figura 6: Diagrama de Fluxo de um sistema DATMO

O módulo de detecção e segmentação leva em conta certas características do sensor, meio ambiente e entidades rodoviárias para extrair parâmetros do objeto a ser rastreado a partir das medições do sensor [31]. Já o módulo de associação visa in-

terligar os objetos observados na varredura atual do sensor aos mesmos objetos observados em varreduras anteriores. Em geral, as formas mais usuais de implementação são uma abordagem probabilística (*Joint Probabilistic Data Association* - JPDA) [33] [34], uma abordagem de rastreamento por múltiplas hipóteses (*Multiple Hypothesis Tracking* - MHT) [21], ou uma abordagem do vizinho mais próximo (*Nearest Neighbor* - NN) [20] [35]. Finalmente, uma vez que as medições estão associadas, um filtro de rastreamento estima os novos parâmetros do objeto. Diferentes filtros podem ser aplicados dependendo se o sistema é descrito como um modelo linear ou um modelo linearizado (tipicamente um filtro de Kalman [26] [31] ou um filtro Kalman estendido [36]), ou como um modelo não linear (tipicamente um filtro de Kalman Unscented [37], ou uma abordagem via método de Monte Carlo, também conhecido como filtro de partículas [17] [23] [38]).

Outras abordagens para a resolução do problema de detecção e rastreamento de objetos móveis aplicados a veículos autônomos também são utilizadas. Uma delas é o DATMO baseado em modelo [17]. Ele funciona através da modelagem direta do sensor físico e da geometria dos objetos móveis com a utilização de um filtro não paramétrico (por exemplo, filtro de partículas). A maior parte do trabalho aqui é feito pelo próprio filtro. O principal desafio é fazer com que o filtro seja eficiente para atender a alta demanda na condução autônoma [15]. Outra abordagem é conhecida como BOF (*Bayesian Occupancy Filtering*). Ela combina uma representação de *grid* de ocupação com técnicas de filtragem Bayesiana. Veja [39] para mais detalhes.

O sistema de DATMO desenvolvido neste trabalho de dissertação é baseado no modelo clássico para a resolução do problema, ao qual daremos maior ênfase a seguir.

2.1 Segmentação da Nuvem de Pontos 3D

Uma vez que os segmentos obtidos por parte do módulo de detecção e segmentação são utilizados no algoritmo na etapa de associação de dados para harmoniza-

ção dos rastreamentos existentes, o desempenho deste módulo é muito significativo para todo o sistema de rastreamento. Para melhorar esta parte, muitos algoritmos para a segmentação têm sido discutidos.

2.1.1 Pré-Processamento de Pontos 3D

Para atingir os objetivos referidos acima, métodos de pré-processamento dos dados do LIDAR 3D são necessários. Uma de suas principais razões é a enorme quantidade de dados fornecida pelo sensor 3D. Um único varrimento do sensor de 3D é geralmente muitas vezes maior do que a de um varrimento 2D, e um problema relacionado com os dados de laser 3D são as camadas inferiores, pois o *scanner* geralmente percebe o solo ou piso, o que faz a interpretação dos dados mais complexa [26].

O solo é geralmente o plano dominante na maioria das cenas na estrada e no meio urbano. Como demonstrado por B. Douillard et al. [40], a extração do solo melhora significativamente o desempenho da segmentação. Geralmente, o algoritmo RANSAC [41] (*Random Sample Consesus*) é utilizado para a remoção do solo [42] [43]. Este algoritmo foi publicado por Fischler e Bolles em 1981. O algoritmo RANSAC assume que o dado analisado é constituído tanto por *inliers* como por valores discrepantes. Os *inliers* podem ser explicados por um modelo com um determinado conjunto de valores como parâmetros, enquanto os *outliers* não se encaixam nesse modelo em qualquer circunstância.

No entanto, de acordo com A. Borcs et al. [44], as nuvens de pontos de representação do solo muitas vezes apresentam diferenças de elevação significativas, devido à inclinação entre os lados opostos da estrada. Nestes casos, utilizando o método RANSAC, a estimativa do solo plano leva a uma segmentação imprecisa, e produz erros significativos nas formas dos objetos extraídos, por exemplo, parte do fundo do objeto pode ser cortada, ou os objetos podem variar ao longo do chão.

Outra abordagem é a utilização de um mapa do tipo *grid* de ocupação [18] [20] [26]. Nos *grids* de ocupação, o ambiente visitado pelo robô é subdividido em um quadriculado composto por células de mesma dimensão. Cada uma dessas células armazena a probabilidade de existir um obstáculo dentro dos seus limites. Um incremento ou decremento da probabilidade é calculado de acordo com um valor que indica se um obstáculo foi detectado ou não pelo sensor.

Após a aplicação do método de remoção de solo, todos os pontos remanescentes são considerados provenientes de obstáculos. Esse método tem se mostrado eficiente na remoção do solo [44] [45] e detecção de objetos, principalmente em um ambiente dinâmico, como em um cenário urbano (Figura 7). Desta forma, esse é o modelo adotado nesta dissertação.

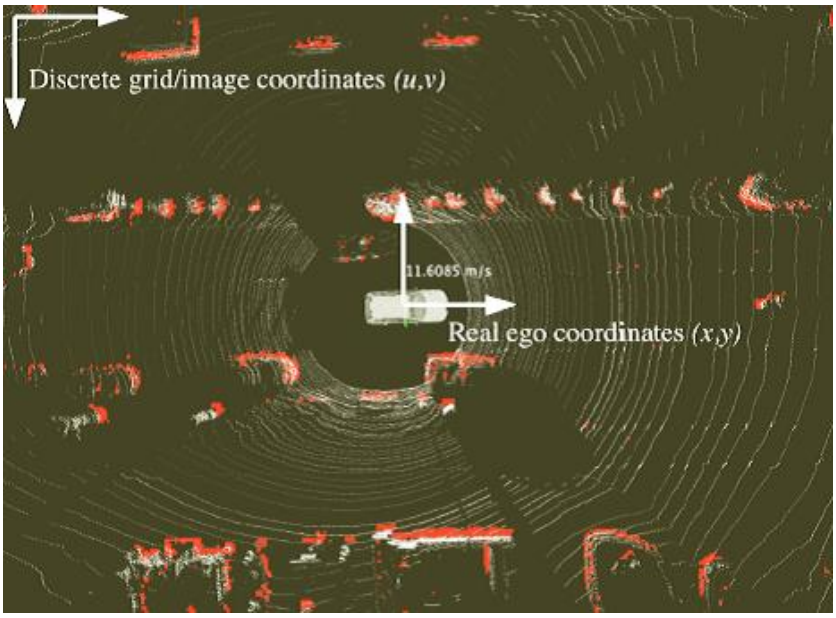


Figura 7: *Grid* de ocupação (com células ocupadas mostradas em vermelho) e a nuvem de pontos completa (em branco) sobrepostos [45]

2.1.2 Segmentação da Nuvem de Pontos 3D

O processo de segmentação de dados visa dividir os pontos de dados coletados em segmentos distintos tais que os pontos associados com o mesmo objeto sejam

agrupados. Como um primeiro critério, um método comumente utilizado para realizar a segmentação é baseado numa distância limiar.

Em [46], os autores afirmam que, com alguma abstração, pode-se pensar em uma nuvem de pontos 3D como um conjunto de pontos de laser 2D dispostos em fatias ou camadas. Assim, técnicas para a detecção de objetos em dados de alcance 2D podem ser estendidas para o caso 3D. Neste trabalho, eles utilizam um algoritmo chamado *Jump Distance Clustering* (JDC), onde o JDC inicializa um novo segmento cada vez que a distância entre dois pontos consecutivos excede um limiar *d*. Veja a Figura 8. Outros exemplos similares a esta abordagem podem ser encontrados nos trabalhos realizados por [20] [21] [47]. Contudo, em [48], os autores afirmam que o algoritmo baseado em limiar de distância entre feixes consecutivos do laser é muito sensível a ruído.

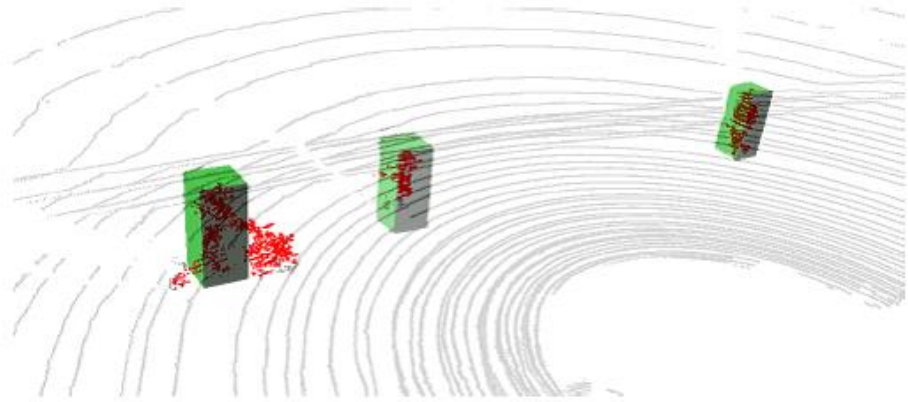


Figura 8: Segmentação e detecção de pedestres em uma nuvem de pontos 3D utilizando o algoritmo *Jump Distance Clustering* (JDC) [46]

Outra abordagem é o uso do algoritmo DBSCAN [49] (*Density-Based Spatial Clustering of Applications with Noise*) [50]. O DBSCAN é baseado no método de detecção de agrupamentos por densidade e formatos arbitrários em conjuntos de dados contendo *outliers*. Clusters são regiões com alta densidade de pontos no espaço dos dados, separadas de outras regiões densas, por regiões de baixa densidade. Um inconveniente do método é o seu requisito de tempo ao tratar com grandes conjuntos de dados [51].

Um simples, mas eficiente método de agrupamento por distância euclidiana é o *Euclidean Cluster Extraction* [52] [53] (Figura 9).

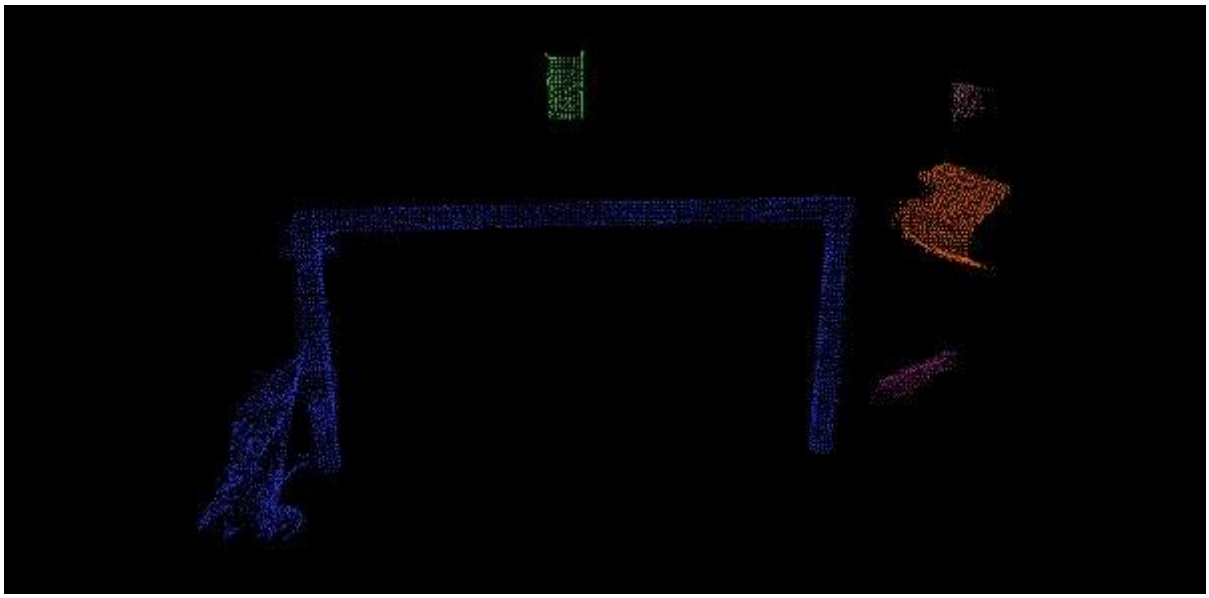


Figura 9: Exemplo de segmentação 3D em uma nuvem de pontos 3D utilizando *Euclidean Cluster Extraction* [52]. Objetos diferentes são destacados por cores diferentes

Esse método tem sido comumente utilizado [54] [55] [56] [57] [58]. Ele é baseado no método hierárquico da distância do vizinho mais próximo. Para isso, calcula-se a matriz de distâncias entre os *n* pontos dos dados 3D, e em seguida os pontos mais próximos são agrupados. É aplicado com uma estrutura de dados de Árvore K-D ou Árvore de Pesquisa Binária Multidimensional e, por conseguinte, é muito eficiente [59]. Neste trabalho é utilizado o *Euclidean Cluster Extraction* para clusterização dos dados oriundos do LIDAR 3D.

2.2 Associação de Segmentos

Uma vez que os dados tenham sido divididos em partes, essas partes precisam ser associadas aos objetos. No rastreamento de objetos, geralmente múltiplas medições aparecem devido aos próprios objetos, como também devido a ruídos de medição. As medições incorretas são referidas como falsas medições. A associação de dados

lida com o problema de selecionar que a medida mais provável é originada a partir do objeto a ser rastreado. Se a medição errada é selecionada, ou se a medição correta não é detectada, as estimativas de estado irão produzir resultados incorretos. A seguir apresenta-se um resumo dos métodos de associação de dados.

2.2.1 *Joint Probabilistic Data Association* (JPDA)

O método *Joint Probabilistic Data Association* (JPDA) é uma abordagem para rastrear vários objetos em movimento. Este método se baseia no modelo Bayesiano onde, se estima a correspondência entre as características detectadas nos sensores e os vários objetos a serem rastreados e forma uma matriz de hipóteses, incluindo todas as associações possíveis. Em outras palavras, calcula as probabilidades de associação do conjunto de medições para os vários objetos em cena. As atribuições com maior probabilidade são escolhidas. Cada objeto deve ter um modelo dinâmico e as medidas devem ser lineares. No caso de um modelo não linear, recomenda-se uma linearização. Uma explicação detalhada pode ser encontrada em [60]. A Figura 10 apresenta um exemplo do uso do método *Joint Probabilistic Data Association* no rastreamento de múltiplos objetos.

Apesar desse método ser amplamente utilizado em abordagens de DATMO [33] [34] [61] [62], ele apresenta algumas desvantagens. Uma desvantagem é que ele considera somente os modelos lineares, e a suposição de linearidade nem sempre é realista. A linearização pode ser feita, mas isso só funcionará se as não linearidades forem fracas na região de interesse. Outra desvantagem é o pressuposto que o número de objetos móveis seja conhecido e constante, o que em um ambiente urbano é difícil prever [63]. Variantes para aplicações não lineares de JPDA têm sido desenvolvidas, mas de acordo com A. Petrovskaya et al. [15], ainda pouco explorados em DATMO para veículos autônomos.

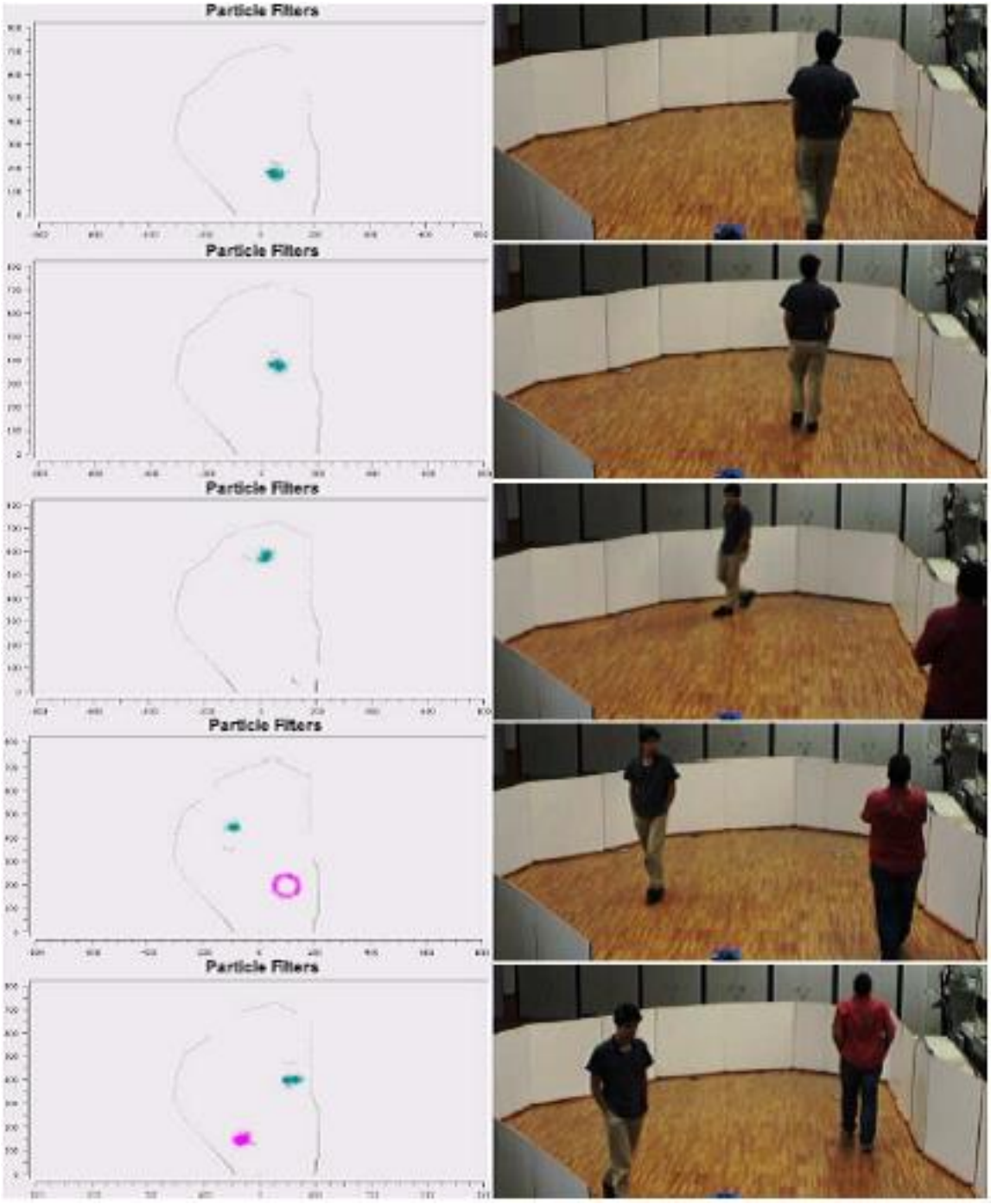


Figura 10: Exemplo do uso do método *Joint Probabilistic Data Association* no rastreamento de múltiplos objetos [34]

2.2.2 *Multiple Hypothesis Tracking* (MHT)

No método de rastreamento por múltiplas hipóteses (MHT), múltiplos alvos também são aceitos. Enquanto o método de associação JPDA assume apenas uma hipótese de associação entre objetos e medidas, o MHT assume várias hipóteses de associação simultâneas e aguarda até que mais informação sensorial do ambiente esteja disponível para julgar quais hipóteses devem ser eliminadas e quais devem ser mantidas. Ou seja, para compor o conjunto de hipóteses utilizam-se também as medições passadas. Novas hipóteses são geradas pela associação dos conjuntos de hipóteses estimadas e as medições recebidas [64]. As possíveis associações que poderão ser realizadas são: se a hipótese é a continuação de um rastreamento já existente; se é um alarme falso e; se é um novo objeto a ser rastreado [65]. Veja [60] e [66] para uma revisão detalhada.

O MHT é usado com sucesso em vários trabalhos de DATMO [21] [67] [68]. Veja a Figura 11.

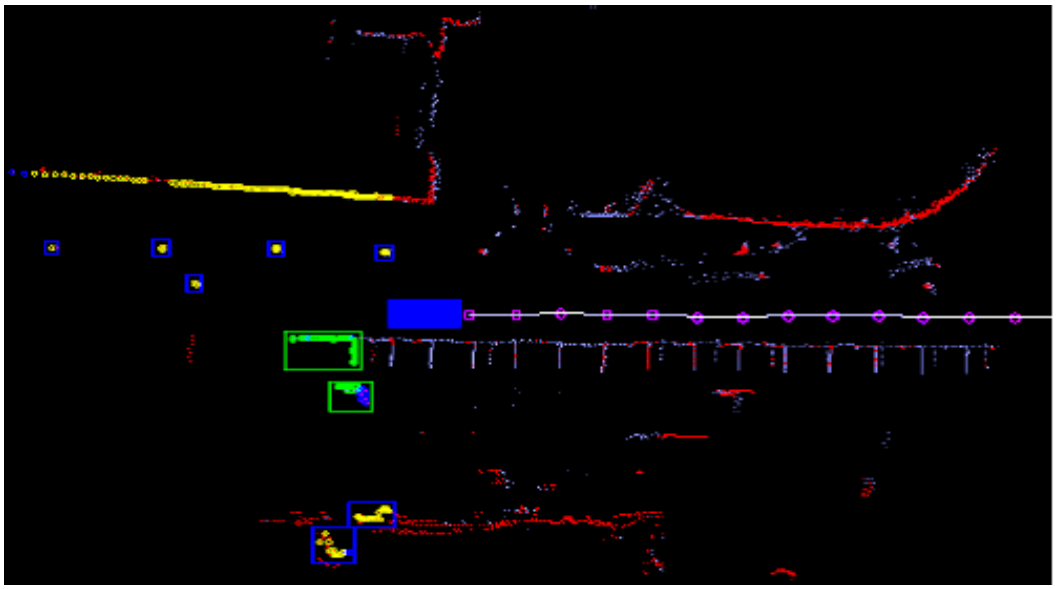


Figura 11: Exemplo do uso do método de associação de dados MHT (*Multiple Hypothesis Tracking*). Veja [21] para mais detalhes

Mas, uma desvantagem dessa abordagem é o seu alto consumo computacional, o que pode ser tornar um impedimento no rastreamento de objetos por períodos longos, já que ele frequentemente gera muitas hipóteses para manter o rastreamento correto. O mesmo efeito é verificado quando muitos objetos precisam ser rastreados simultaneamente. Para permitir uma aplicação em tempo real do MHF, o tamanho da árvore de hipóteses deve ser limitado, por exemplo, por um algoritmo de poda (*pruning algorithm*) [60].

2.2.3 *Nearest Neighbor* (NN)

Outro método amplamente utilizado é do vizinho mais próximo (*Nearest Neighbor* – NN) [20] [35] [69] [70]. O NN busca associar um dado objeto à medida mais próxima, segundo um critério de distância, dentre as medidas associáveis ao rastreamento. A Figura 12 apresenta um exemplo da utilização do método NN na associação de objetos em movimento.

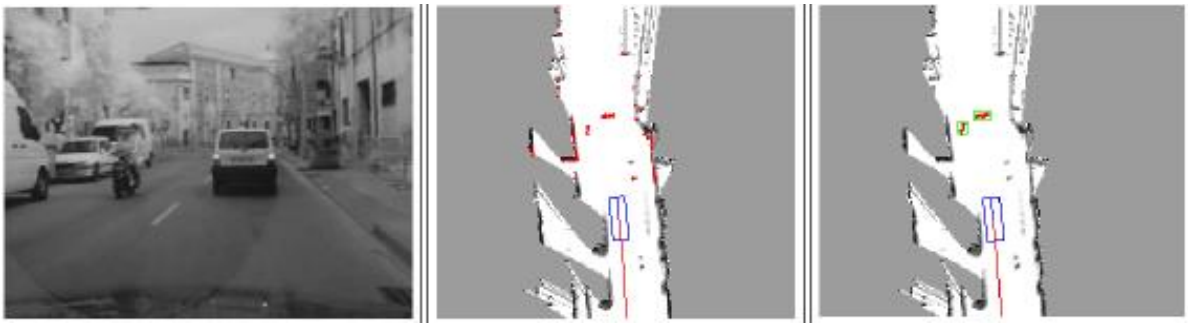


Figura 12: Exemplo de detecção de objetos móveis com a associação de dados pelo vizinho mais próximo [20]

Para que não ocorram situações nas quais mais de um rastreamento seja associado à mesma medida o algoritmo garante que, ao final do processo de associação, não exista mais de um rastreamento associado a uma mesma medida. Medidas associadas são utilizadas para atualizar o rastreamento. Medidas não associadas são usadas para gerar novos rastreamentos e rastreamentos não associados são mantidos

para a próxima iteração até um limiar pré-estabelecido. Leia [60] e [63] para informações detalhadas.

Uma desvantagem do NN é que associações erradas se mantêm e o impacto dessas associações errôneas pode prejudicar o desempenho do algoritmo. Sua vantagem é a simplicidade de implementação e a baixa complexidade computacional [69]. Uma abordagem semelhante a este método foi desenvolvida nesta dissertação.

2.3 Rastreamento de Objetos em Movimento

Como descrito anteriormente, diversas abordagens são utilizadas como filtro para o rastreamento de objetos, sendo o Filtro de Kalman e o Filtro de Partículas as duas principais formas mais amplamente empregadas. Ambas são baseadas no filtro de Bayes que é uma abordagem probabilística para estimar uma função de densidade de probabilidade recursivamente ao longo do tempo utilizando medições de entrada e um modelo de processo matemático.

Na Seção 2.3.1 é apresentada uma breve descrição do Filtro de Kalman. Já na Seção 2.3.2 uma descrição um pouco mais detalhada do Filtro de Partículas é demonstrada, por se tratar da abordagem adotada neste trabalho.

2.3.1 Filtro de Kalman

O Filtro de Kalman é um método matemático criado por Rudolf Kalman [71]. É um filtro recursivo idealmente projetado para estimar o estado de um sistema dinâmico de múltiplas medições sequenciais. O Filtro de Kalman é geralmente aplicado nos modelos de sistemas lineares discretizados no domínio do tempo. A cada passo de tempo, um operador linear é aplicado ao estado para gerar o próximo estado, com adição de ruído e, opcionalmente, alguma informação sobre as entradas de controle, caso sejam conhecidas.

O estado é estimado pelas equações de forma recursiva. O Filtro de Kalman assume um modelo de sistema dinâmico, conforme descrito abaixo:

$$x_k = Ax_{k-1} + \mathrm{B}u_{k-1} + w_{k-1} \quad (1)$$

Onde, x_k denota um estado de sistema no tempo discreto *k*, *A* é uma matriz de estado, B uma matriz opcional de controle e u_k é uma entrada de controle. A equação de medição é dada por:

$$y_k = Cx_k + u_k \quad (2)$$

Onde, y_k é a medição, e C uma matriz que descreve o estado para a medida no tempo *k*. w_k ~ *N* (0, *Q*) e u_k ~ *N* (0, *R*) são ruídos gaussianos de média zero e covariância *Q* e *R*. Essas variáveis representam os ruídos de estado e das medidas (respectivamente) e assume-se serem independentes uma da outra.

As equações de atualização dos estados podem ser agrupadas em dois tipos distintos: equações de atualização do tempo e equações de atualização da medição [65].

As equações de atualização do tempo são responsáveis pelo avanço das variáveis de estado e das covariâncias no tempo para se obter as estimativas anteriores (*a priori*) para o próximo instante:

$$\hat{x}'_k = A\hat{x}_{k-1} + Bu_k \quad (3)$$

$$\hat{P}'_k = AP_{k-1}A^T + Q \quad (4)$$

As equações de atualização das medições são responsáveis pela retroalimentação, incorporando uma nova informação da variável observável nas estimativas anteriores para obter um ganho (ou melhoria) na estimação posterior.

$$K_k = \hat{P}'_k C^T [C\hat{P}'_k C^T + R]^{-1} \quad (5)$$

$$\hat{x}_k = \hat{x}'_k + K_k[y_k - Cx_k'] \quad (6)$$

$$P_k = [I + K_k C]P_k' \quad (7)$$

Onde, K_k representa o ganho de Kalman ou o fator de combinação que minimiza a covariância do erro a *posteriori* e P_k a covariância do erro estimando a *posteriori*. Por ser um clássico método de filtragem em processamento de sinal, o Filtro de Kalman tem sido amplamente aplicada em sistemas de rastreamento baseados em laser [19] [20] [21] [26] [31] [32] [35] [36] [45].

Numa das abordagens, Wang, et al. propuseram uma possível solução para o problema de mapeamento e localização simultâneos (SLAM) juntamente com o problema de detecção e rastreamento de objetos em movimento (DATMO) [21]. Abordagem semelhante foi proposta por T.-D. Vu et al. [20]. A vantagem dessa abordagem é que o mapa de SLAM pode ajudar a detectar objetos em movimento, e por sua vez, o mapa será mais confiável após a remoção dos objetos em movimento detectados e monitorados pela DATMO. Veja a Figura 13.

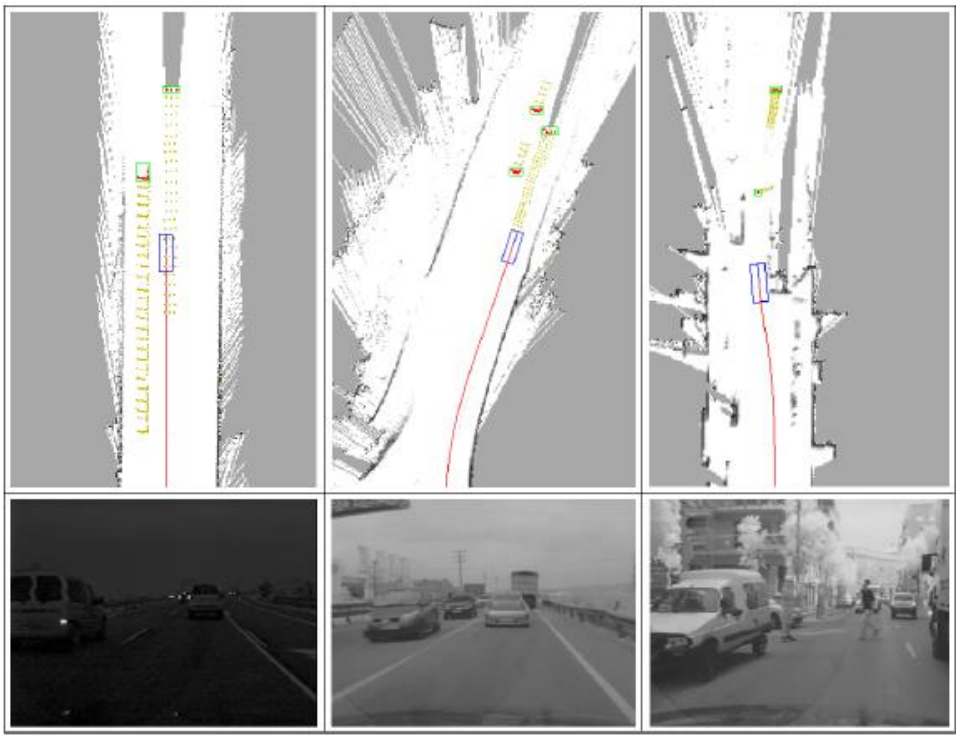


Figura 13: Resultados experimentais de SLAM e DATMO em tempo real para diferentes ambientes [20]

Um filtro de Kalman também foi aplicado para o monitoramento de pedestres em [35]. No método de detecção e rastreamento de objetos em movimento proposto por Wu e Sun [36], é utilizada a combinação de um mapa de grade de ocupação (*occupancy grid map*) e um Filtro de Kalman estendido (EKF, do inglês *Extended Kalman Filter*).

O filtro de Kalman pode ser visto como um caso particular em que o modelo é linear e as distribuições das variáveis de estado e observação são Gaussianas. Entretanto, nem sempre os fenômenos podem ser tratados por modelos lineares [72]. A não linearidade e a não gaussianidade são propriedades comumente encontradas e os métodos Sequenciais de Monte Carlo (SMC) destacam-se com vantagens sobre os demais no que se refere ao cálculo da distribuição *a posteriori* nessas ocasiões [73]. Veja a Seção seguinte para mais detalhes.

2.3.2 Filtro de Partículas

Os algoritmos SMC apareceram na literatura sob diversas denominações: filtros *bootstrap*, condensação, filtro de partículas, filtro de Monte-Carlo [63] [72]. O Filtro de Partículas (FP) têm recebido crescente atenção como um método alternativo para aproximar a solução do problema de inferência recursiva de estados ocultos em sistemas dinâmicos não lineares e não gaussianos [14] [74] [75] [76] [77].

O FP consiste na utilização de uma amostra gerada aleatoriamente com probabilidade de observação conhecida para aproximar o valor da função de interesse através de seu valor estimado. A probabilidade *a posteriori* é representada não mais por uma Gaussiana, mas por uma distribuição discreta de probabilidade definida através de um conjunto amostral e seus respectivos pesos. Veja Figura 14. Cada amostra desse conjunto é denominada *partícula*. Essa representação é aproximada, mas é não paramétrica, e, portanto, podem representar um espaço muito mais amplo do que as distribuições gaussianas [14].

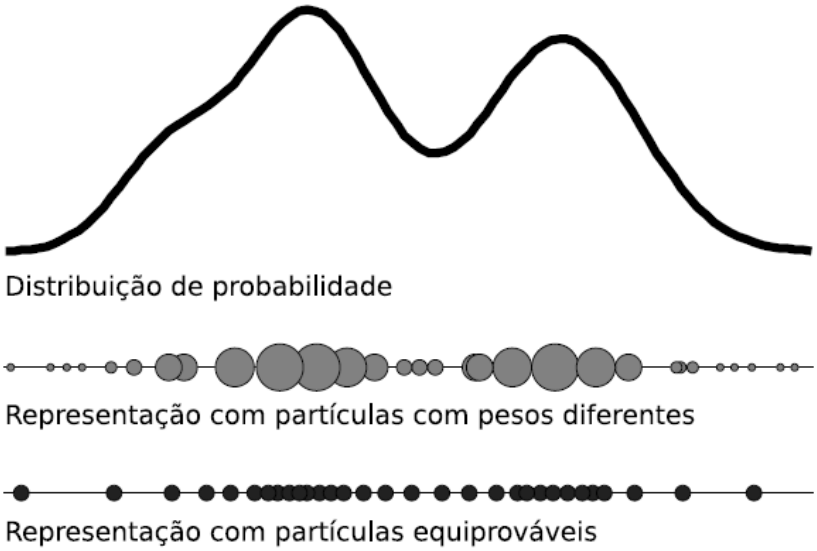


Figura 14: Representação de uma distribuição de probabilidades através de partículas. A distribuição acima pode ser representada por partículas com peso (centro) ou sem peso (abaixo), neste último caso a concentração de partículas é o único indicador da probabilidade [78]

Considerando o modelo de espaço de estado não linear descrito por:

$$X_k = f(X_{k-1}, V_{k-1}) \tag{8}$$

$$Y_k = g(X_k, U_k), \tag{9}$$

onde X_k é a variável de estado no instante de tempo *k*, Y_k são as observações no instante de tempo *k*. As funções *f* e *g* são não lineares e geram X_k e Y_k. Temos ainda que V_k é o ruído do sistema e U_k é o ruído da medição. Os ruídos podem ser assumidos como variáveis aleatórias com distribuição normal e covariância Q_k e R_k.

A função de densidade de probabilidade $P(X_k | X_{k-1})$ é referente à transição ou predição da distribuição no instante *k*. É conhecida como função de transição de estado e é definida pela dinâmica do sistema (Equação 8). Já a Equação (9) implica a função de densidade de probabilidade $P(Y_k | X_k)$, que representa a observação.

A questão consiste em obter a melhor estimação para a variável de estado X_k quando apenas dados das observações estão disponíveis. Assumindo que X_k segue um

processo de Markov, ou seja, o estado atual pode ser determinado apenas pelo estado anterior, a distribuição *a posteriori* pode ser definida como $P(X_k|Y_{1:k})$.

Através da função densidade do estado inicial X_0 do sistema, em conjunto com uma fase de transição e uma de observação, é possível chegar à função densidade no tempo seguinte. Assim, a função densidade *a posteriori* $P(X_k|\, Y_{1:k})$ pode ser avaliada de forma recursiva.

Propagando a fase de transição, de tempo k, por meio da densidade de transição dada por:

$$P(X_k|Y_{1:k-1}) = \int P(X_k|\, X_{k-1})P(X_{k-1}|\, Y_{1:k-1})dX_{k-1}\,, \tag{10}$$

e a fase de observação, pela aplicação do teorema de *Bayes* [76], quando novos dados chegam, de acordo com:

$$P(X_k|Y_{1:k}) = \frac{P(Y_k|\, X_k)P(X_k|\, Y_{1:k-1})}{\int P(Y_k|\, X_k)P(X_k|Y_{1:t-1})dX_k}, \tag{11}$$

é possível fornecer a solução ótima para o problema de estimação, mas infelizmente o cálculo da distribuição posterior e dos estimadores Bayesianos são proibitivamente complexos [72]. Para transpor esta dificuldade o filtro de partículas adota uma abordagem baseada em simulação cuja uma das técnicas básicas é denominada amostragem por importância sequencial (*Sequential Importance Sampling* - SIS). O objetivo é estimar a densidade de probabilidade posterior e a ideia central do filtro de partículas é representar tais densidades por conjunto de partículas.

2.3.2.1 *Sequential Importance Sampling* - SIS

O algoritmo SIS é um método de Monte Carlo (MC), que constitui a base para a maioria dos filtros MC sequenciais desenvolvidos nas últimas décadas. Veja [75].

Neste filtro, as distribuições de probabilidade são representadas por um conjunto de partículas $X_k = \{x_k^{(1)}, x_k^{(2)}, \dots, x_k^{(N)}\}$. Cada partícula $x_k^{(i)} \in \mathcal{R}^d$ representa, na iteração *k*, uma hipótese e possui um peso associado $w_k^{(i)}$, de forma que $\sum_i w_k^{(i)} = 1$.

O filtro atualiza a distribuição de partículas em duas etapas: uma etapa de predição e uma etapa de atualização. Iniciando o sistema com uma distribuição $P(x_{k-1}|Y_{k-1})$ formada de partículas com mesmo peso. Y são as observações.

Na etapa de predição, cada partícula $x_{k-1}^{(i)}$ de $P(x_{k-1}|Y_{k-1})$ é movida, de forma independente, para uma nova posição no espaço de estado $x_k^{(i)}$ com probabilidade $P(x_k|x_{k-1})$ dada pela função de transição do sistema. Ou seja, cada partícula passa pela função do sistema $x_k^{(i)} = f_{k-1}(x_{k-1}^{(i)}, V_{k-1})$, onde V_{k-1} representa um ruído de média zero, usado para gerar pequenas perturbações e evitar o colapso das partículas. Em outras palavras, evitar que todo o conjunto se concentra (entre em colapso) em uma única hipótese. Desta forma, é determinada a distribuição *a priori* aproximada $P(x_k|Y_{k-1})$, equivalente à necessária para a aplicação do teorema de Bayes.

Na atualização, quando a distribuição por importância é a distribuição anterior (*a priori*), temos a amostragem por importância sequencial e neste caso cada partícula da distribuição *a priori* tem seu peso, ou probabilidade de ocorrência de um estado, aproximado na forma [75]:

$$w_k^{(i)} \propto P(Y_k | x_k^{(i)}) \tag{12}$$

Sendo os pesos normalizados:

$$\widetilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^{N} w_k^{(j)}} \tag{13}$$

Ainda de acordo com [75], é possível aproximar a probabilidade condicional *a posteriori* e aplicar essa aproximação à equação do valor estimado na forma:

$$\hat{x}_k = \int x_k \sum_{i=1}^{N} \widetilde{w}_k^{(i)} \, \delta\,(x_k - x_k^{(i)}) dx_k \;\approx\; \sum_{i=1}^{N} \widetilde{w}_k^{(i)} \, x_k^{(i)} \tag{14}$$

O efeito dos pesos é tornar mais prováveis as partículas em regiões que correspondem com as observações e, como consequência, formar a distribuição *a posteriori* $P(x_k|Y_k)$ [79]. Ou seja, representa a probabilidade do estado real estar no estado representado por essa amostra.

O problema do algoritmo SIS é que com o transcorrer do tempo o sistema degenera-se rapidamente. Em outras palavras, a variância incondicional dos pesos de importância aumenta com o tempo. Veja [74] [75] para detalhes. A solução adotada para evitar este problema foi criar uma etapa adicional. Trata-se da etapa de seleção ou de re-amostragem.

2.3.2.2 *Sampling Importance Resampling* - SIR

O algoritmo Amostrando e Re-amostrando pela Importância (*Sampling Importance Resampling* - SIR) tenta contornar o problema da degeneração das partículas do Filtro de Partículas re-amostrando as que contribuem pouco para a solução [75].

Uma vez calculados os pesos por importância, antes de fazer a evolução para o instante seguinte, é realizada a seleção. As partículas de maior peso (maior importância) são selecionadas. De acordo com o seu peso é realizada uma nova amostragem (da distribuição anterior). Assim, as partículas de maior importância dão origem

a um maior número de partículas. As partículas de menor importância desaparecem e não originam “descendentes” [14] [72].

A Figura 15 mostra as partículas como círculos e seus pesos são representados pelo tamanho dos círculos, onde os círculos com tamanho maior correspondem a pesos de maior importância. De forma análoga, os círculos menores representam pesos de menor importância. A importância de cada partícula é calculada através da densidade *a posteriori* da partícula.

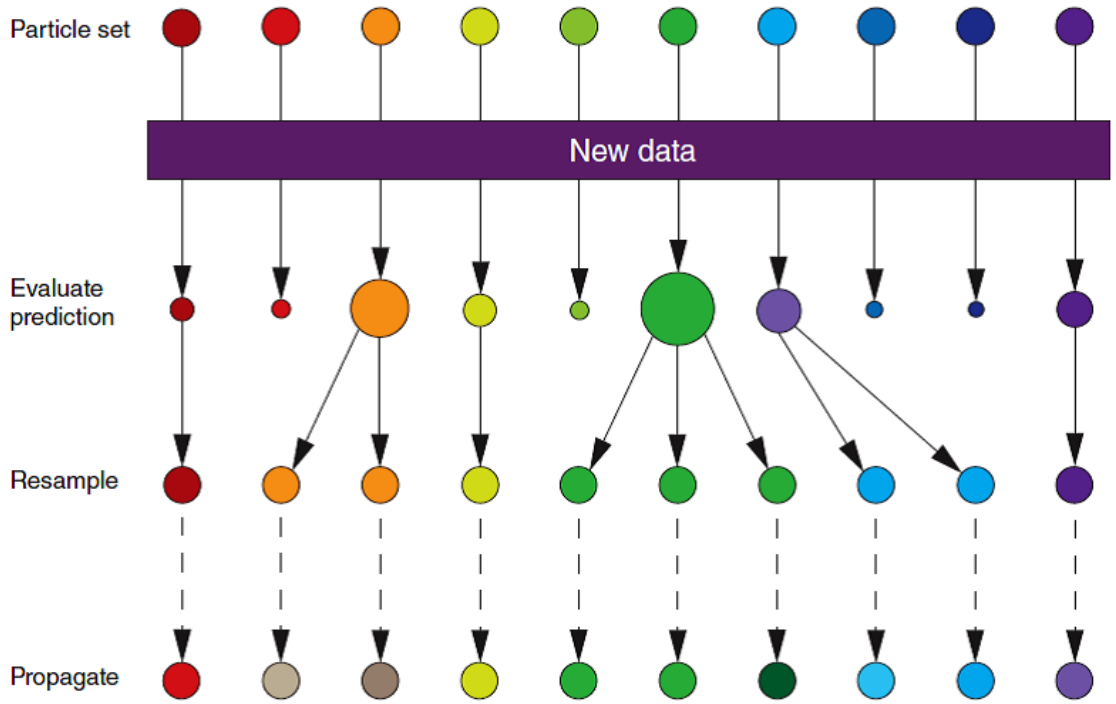


Figura 15: Ilustração de uma iteração do algoritmo *Sampling Importance Resampling*, mostrando partículas com diferentes pesos [80]

Está é a versão do filtro conhecida como *Bootstrap* [74] e já foi utilizada com sucesso no rastreamento de objetos em movimento em [34] [81]. Em [34], os autores utilizaram um laser 2D para sensoriar o ambiente e um JPDA em conjunto com um filtro de partículas (*sample based joint probabilistic data association filters*) para associar os dados e rastrear os objetos.

Um inconveniente dos filtros de partícula é conhecido como a maldição da dimensionalidade [15]. Aplicações que precisam ser modeladas com espaços de hipóteses de alta dimensão sofrem com o aumento exponencial no número de partículas necessário para aproximar adequadamente a distribuição *a posteriori* [72]. Contudo, devido ao aumento do poder computacional ocorrido nos últimos anos, o filtro de partículas tem sido largamente utilizado em sistemas de rastreamento [17] [23] [34] [38] [82] [81] [83] [84] [85].

Em [84], os autores utilizam uma câmera de visão estéreo para detecção e rastreamento de pedestres. O método proposto detecta os movimentos harmônicos dos membros e do corpo de um humano durante uma caminhada e propagam a posição, direção e fase da passada com a utilização de um filtro de partículas. Também utilizando dados fornecidos por um sensor de visão estereoscópica, R. Danescu e S. Nedevschi [85] propõem uma abordagem para modelar o ambiente dinâmico utilizando um mapa de elevação dinâmico (*Dynamic Digital Elevation Maps*). O mapa é representado por uma população de partículas, tendo cada partícula uma posição, uma altura, e uma velocidade. A fim de descrever corretamente obstáculos que se deslocam, uma distribuição de probabilidade baseada na velocidade é utilizada.

Visto o bom desempenho dos filtros de partícula para o problema de inferência de estados ocultos em sistemas dinâmicos não lineares e não gaussianos, neste trabalho um filtro *Bootstrap* foi utilizado para estimativa do estado (posição, orientação e velocidade) dos veículos em movimento ao redor do automóvel robótico autônomo.

3 SISTEMA DE DETECÇÃO E RASTREAMENTO DE VEÍCULOS EM MOVIMENTO

Este capítulo apresenta o sistema de DATMO proposto neste trabalho para detecção e rastreamento de veículos em movimento. A Seção 3.1 traz uma visão geral do sistema, apresentando em alto nível as etapas em que ele opera. As Seções 3.1 a 3.4 detalham estas etapas.

3.1 Visão Geral

A Figura 16 mostra o diagrama de fluxo do sistema de DATMO proposto neste trabalho.

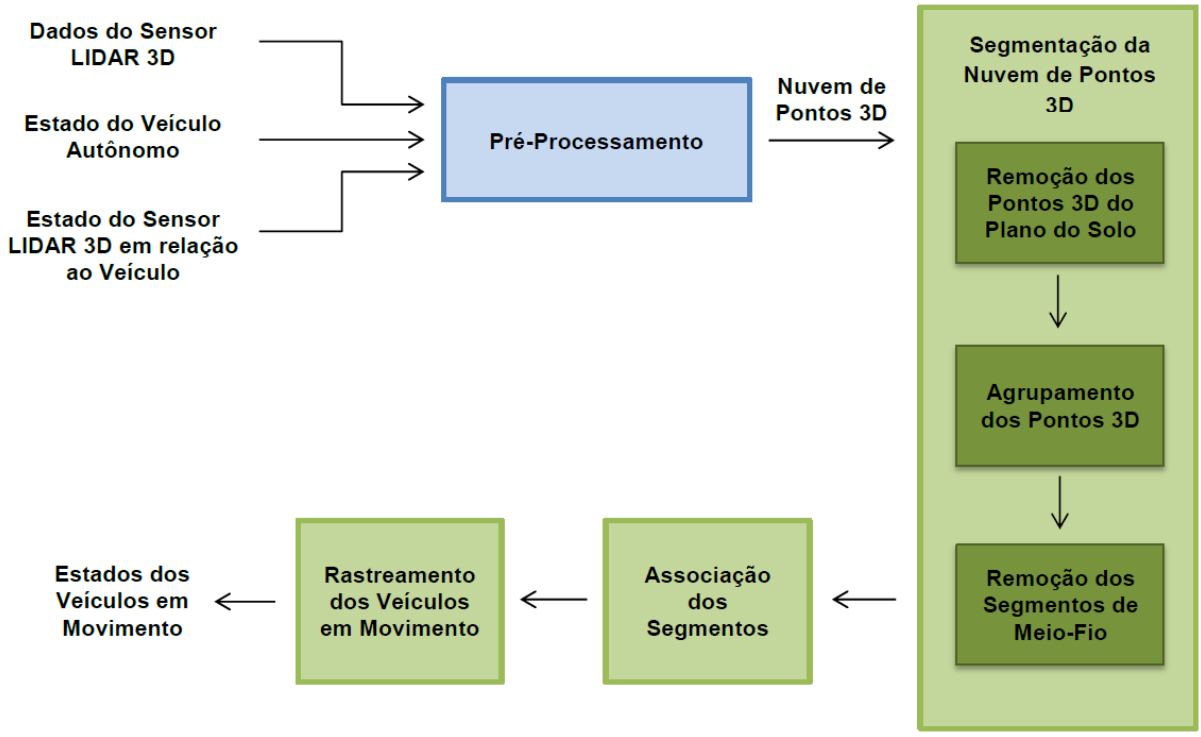


Figura 16: Diagrama de fluxo do sistema de DATMO proposto

O sistema de DATMO proposto opera em três etapas: segmentação, associação e rastreamento. A cada varredura do sensor LIDAR 3D, após a conversão dos dados do sensor em uma nuvem de pontos 3D com coordenadas globais, na etapa de segmentação, os pontos associados ao plano do solo são removidos; a nuvem de pontos é segmentada em agrupamentos de pontos usando a distância Euclidiana, sendo que cada agrupamento representa um objeto no ambiente; e os agrupamentos relacionados aos meios-fios são removidos. Na etapa de associação, os objetos observados na varredura atual do sensor são associados aos mesmos objetos observados em varreduras anteriores usando o algoritmo do vizinho mais próximo (*nearest neighbor*). Finalmente, na etapa de rastreamento, os estados dos objetos são estimados usando um filtro de partículas. Os objetos com velocidade acima de um determinado limiar são considerados veículos em movimento.

3.2 Segmentação da Nuvem de Pontos 3D

Antes da segmentação da nuvem de pontos 3D, é necessária uma fase de pré-processamento para converter os dados do sensor LIDAR 3D em uma nuvem de pontos com coordenadas globais (Figura 16). Para esta conversão, é necessário o conhecimento do estado (posição e orientação) do veículo autônomo, bem como a posição e orientação do sensor no veículo. Esta fase de pré-processamento não será tratada neste trabalho, dado que um sistema eficiente para computação da nuvem de pontos com coordenadas globais já foi implementado pelo LCAD, ao qual este trabalho está associado. Sendo assim, utilizou-se a nuvem de pontos computada pelo sistema do LCAD a cada varredura do sensor.

Cada ponto p_i da nuvem de pontos é representado por uma coordenada global (x_i, y_i, z_i). O componente z_i é ortogonal ao plano xy e representa a altura de p_i. A Figura 17 mostra um exemplo de uma nuvem de pontos com coordenadas globais do ambiente trafegado pelo veículo autônomo. Na Figura 17, os pontos destacados em azul possuem $z_i > 0$ e aqueles destacados em vermelho possuem $z_i \leq 0$.

Figura 17: Exemplo de uma nuvem de pontos 3D com coordenadas globais do ambiente trafegado pelo veículo autônomo. Os pontos destacados em azul possuem $z_i > 0$ e aqueles destacados em vermelho possuem $z_i \leq 0$

3.2.1 Remoção dos Pontos 3D do Plano do Solo

O solo é geralmente o plano dominante na maioria das cenas em estradas e em meios urbanos. A remoção do plano do solo melhora significativamente a segmentação da nuvem de pontos em agrupamentos de pontos que representam objetos no ambiente [40].

Geralmente, o algoritmo RANSAC (Seção 2.1.1) é usado para remover o plano do solo. Entretanto, em nossos experimentos em um ambiente urbano dinâmico usando um sensor LIDAR 3D, o algoritmo não obteve os resultados esperados, falhando em remover pontos associados ao plano do solo ou até mesmo eliminando pontos relevantes, como, por exemplo, pontos relacionados a parte de baixo ou a roda de veí-

culos a serem rastreados. A Figura 18 mostra um exemplo de uma nuvem de pontos após a execução do algoritmo RANSAC para remoção do plano do solo. Na Figura 18, os pontos destacados em branco são associados aos objetos considerados obstáculos e aqueles destacados em vermelho são associados ao plano do solo e não removidos pelo algoritmo.

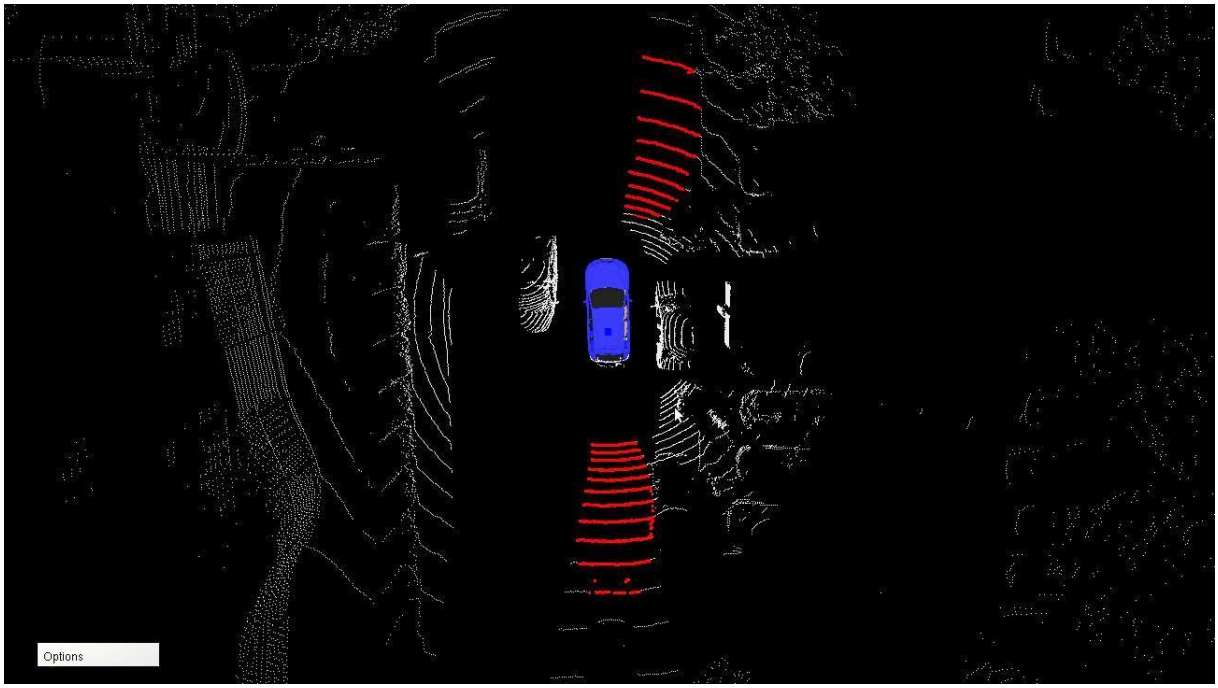


Figura 18: Exemplo de uma nuvem de pontos 3D após a execução do algoritmo RANSAC para remoção do plano do solo. Os pontos destacados em branco são associados aos objetos considerados obstáculos e aqueles destacados em vermelho são associados ao plano do solo e não removidos pelo algoritmo

Uma abordagem para remoção do plano do solo usando um sensor LIDAR 3D já desenvolvida pelo LCAD [86], ao qual este trabalho está associado, se mostrou mais eficiente e foi adotada neste trabalho. Abordagens semelhantes foram usadas por outros trabalhos da literatura [18] [20] [26] [44] [45] (Seção 2.1.1).

A abordagem adotada para remoção do plano do solo utiliza um mapa de *grid* de ocupação. Um *grid* de ocupação representa o mapa do ambiente ao redor do veículo autônomo por um quadriculado composto por células associadas a regiões igualmente espaçadas. Cada uma dessas células armazena a probabilidade de ocupa-

ção, i.e., a probabilidade de existir um obstáculo dentro dos limites de sua região. Geralmente, os algoritmos que usam *grids* de ocupação representam a probabilidade de ocupação por *log-odds*, a fim de evitar instabilidades numéricas nos casos de probabilidades próximas de um ou de zero. O *log-odds* (*L*) é obtido a partir da probabilidade de ocupação (*P*), de acordo com a seguinte equação [14]:

$$L = \log\left(\frac{P}{1-P}\right). \tag{15}$$

Inicialmente, as células do *grid* de ocupação recebem uma probabilidade neutra L_0. Esta probabilidade informa que as células podem estar tanto ocupadas quanto livres. A partir daí, sempre que novos dados do sensor são recebidos, as células no campo perceptual do veículo autônomo são atualizadas por um incremento ou decremento da probabilidade de estarem ocupadas, segundo a seguinte equação [14]:

$$L_T = L_{T-1} + \Delta L - L_0, \tag{16}$$

onde L_T é o *log-odds* de uma determinada célula no instante *T*, L_{T-1} é o *log-odds* da célula no instante *T–1* e ΔL é o incremento ou decremento do *log-odds* calculado de acordo com a probabilidade de um obstáculo ter sido detectado ou não pelo sensor.

Neste trabalho, para detectar obstáculos, as distâncias medidas por cada feixe de laser do sensor LIDAR 3D, começando pelo mais baixo, são comparadas com aquelas medidas pelo feixe de laser localizado imediatamente acima [86]. Maiores detalhes sobre o sensor LIDAR 3D podem ser encontrados na Seção 4.2. A Figura 19 mostra um exemplo de detecção de um obstáculo por meio da comparação das medidas realizadas por dois feixes de laser consecutivos de um sensor LIDAR 3D. Se o veículo autônomo estiver navegando em um plano de solo onde não existem obstáculos em seu campo perceptivo, ambos os feixes de laser irão medir a distância do sensor até o plano. Se existir um obstáculo, as medidas realizadas por um ou ambos os feixes de laser serão menores que a distância do sensor até o plano.

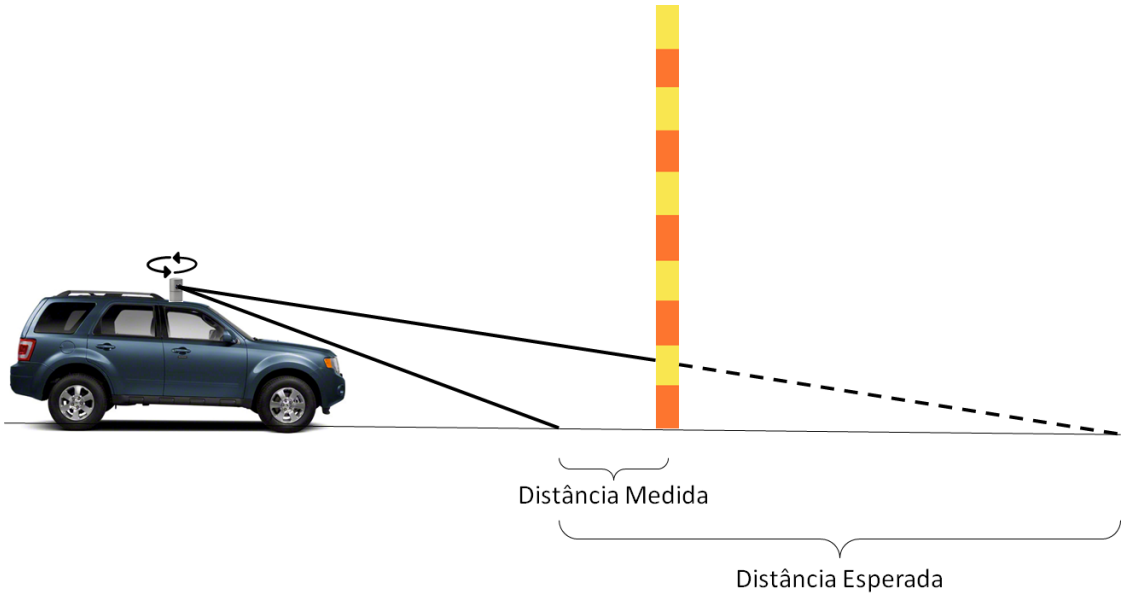


Figura 19: Exemplo de detecção de um obstáculo por meio da comparação das medidas realizadas por dois feixes de laser consecutivos de um sensor LIDAR 3D [86]

Para calcular a probabilidade de um obstáculo ter sido detectado, os dois feixes são projetados no plano do solo e a diferença entre os seus tamanhos é computada. Se o veículo autônomo estiver navegando em um plano sem obstáculos em seu campo perceptivo, esta diferença será igual a um valor esperado – “Distância Esperada” na Figura 19. Se existir um obstáculo, a diferença será menor que o valor esperado – “Distância Medida” na Figura 19. A probabilidade de um obstáculo ter sido detectado é calculada pela equação:

$$P_O = \frac{D_E - D_M}{D_E}, \tag{17}$$

onde D_E é a distância esperada e D_M é a distância medida. O *log-odds* de P_O é usado como incremento ou decremento para atualizar as células tocadas pelo sensor. Maiores detalhes sobre a computação de D_E e D_M são descritos por F. W. Mutz [86].

A Figura 20 mostra a mesma nuvem de pontos da Figura 18 após a execução do algoritmo adotado para remoção do plano do solo. Na Figura 20, os pontos destacados em branco são associados aos objetos considerados obstáculos. Como pode

ser observado ao comparar a Figura 18 e a Figura 20, houve uma melhoria na remoção do plano do solo.

Figura 20: Exemplo de uma nuvem de pontos 3D após a execução do algoritmo adotado para remoção do plano do solo. Os pontos destacados em branco são associados aos objetos considerados obstáculos

Copas de árvores podem ser frequentemente confundidas com objetos em movimento [22]. Para lidar com este problema, uma altura máxima de leitura do sensor é definida e apenas os pontos abaixo desta altura são levados em consideração na detecção e remoção dos pontos do plano do solo e na segmentação da nuvem de pontos. Para evitar remoção de planos diferentes do solo, como, por exemplo, o capô ou o teto de veículos a serem rastreados, uma altura mínima de leitura do sensor é também definida.

3.2.2 Agrupamento dos Pontos 3D

Para a segmentação da nuvem de pontos em agrupamentos de pontos que representam objetos no ambiente ao redor do veículo autônomo, neste trabalho, utilizou-

se um método simples, mas eficiente, de agrupamento por distância Euclidiana, denominado, *Euclidean Cluster Extraction* [54] [55] [56] [58]. Este método é baseado na abordagem hierárquica da distância do vizinho mais próximo. Inicialmente, o algoritmo deste método cria uma estrutura de dados de Árvore K-D para a entrada dos pontos de uma nuvem, P, uma lista vazia de agrupamentos, C, e uma lista dos pontos que precisam ser verificados, Q. Em seguida, para cada ponto $p_i \in P$, o algoritmo adiciona p_i a Q e procura os elementos do conjunto P_k^i dos k pontos vizinhos mais próximos de p_i em uma esfera com raio $r < s_{limiar}$, onde s_{limiar} é um parâmetro do algoritmo. Como medida de proximidade, utiliza-se a distância Euclidiana entre os pontos. Para cada um dos *k* vizinhos $v_i \in P_k^i$, o algoritmo verifica se v_i já foi processado (i.e., encontrado como um vizinho de um outro ponto $p_i \in P$); caso contrário, adiciona v_i a Q. Quando a verificação de todos os pontos em Q é finalizada, o algoritmo adiciona Q à lista de agrupamentos C e redefini Q como uma lista vazia. O algoritmo termina quando todos os pontos $p_i \in P$ foram processados e fazem parte da lista de agrupamentos C [52].

A Figura 21 mostra um exemplo de uma nuvem de pontos após sua segmentação. Na Figura 21, os pontos associados a diferentes agrupamentos (ou segmentos), que representam diferentes objetos no ambiente, são destacados por cores diferentes.

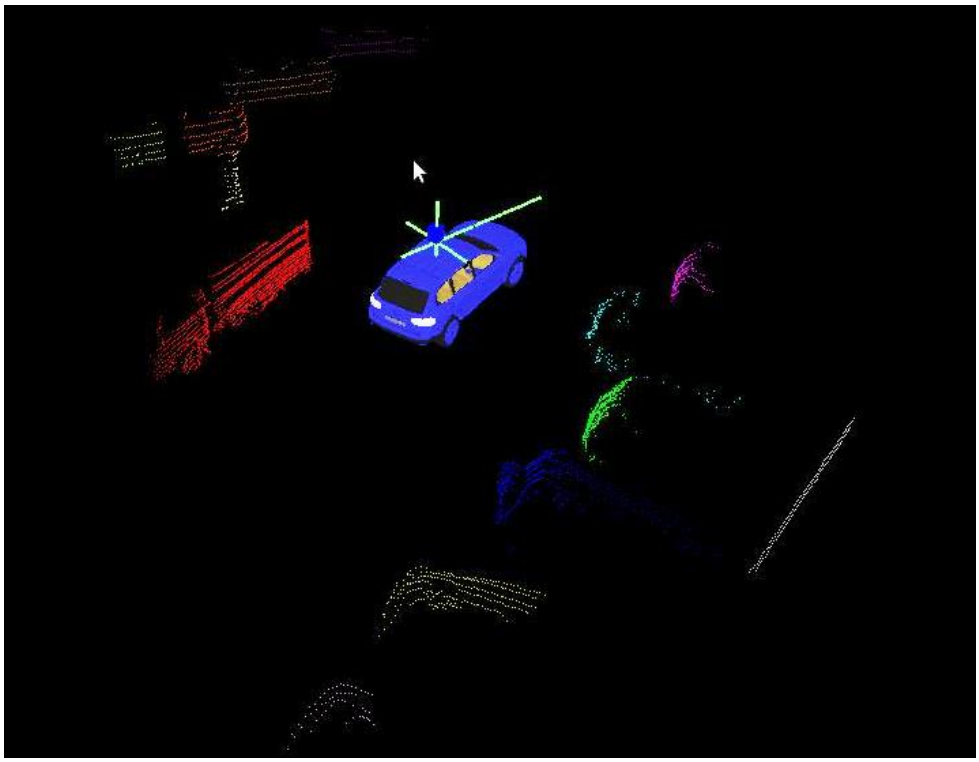


Figura 21: Exemplo de uma nuvem de pontos 3D após sua segmentação. Pontos associados a diferentes segmentos, que representam diferentes objetos no ambiente, são destacados por cores diferentes

Após a segmentação da nuvem de pontos, cada objeto o_j no ambiente é descrito por um segmento $c_j \in C$ e um carimbo do tempo t_j da varredura do sensor na qual ele foi observado, e cada ponto $p_j \in c_j$ é representado por uma coordenada global (x_j, y_j, z_j) (Seção 3.2). Cada objeto é inserido em uma lista de objetos denominada $ListaAtualObjetos = \{o_1, o_2, \ldots, o_M\}$, onde M é o número de objetos observados.

3.2.3 Remoção dos Segmentos do Meio-Fio

Meios-fios podem ser frequentemente confundidos com objetos em movimento. Para tratar este problema, segmentos de meio-fio são detectados e removidos. Os métodos de detecção de meio-fio são divididos basicamente em duas categorias [87]: a primeira é baseada na detecção das características geométricas do meio-fio e a segunda é baseada no contexto da imagem derivado a partir do campo de visão mo-

nocular. O principal mérito de métodos da segunda categoria é que eles contêm informações mais ricas, tais como cor e textura. No entanto, métodos da segunda categoria têm problemas para lidar com ambientes em condições ruins, tais como iluminação deficiente e faixas rodoviárias insuficientes. Em comparação com a segunda categoria, em geral, os métodos da primeira categoria não são afetados pelas condições mencionadas acima e são adaptáveis e robustos em ambientes em condições variadas. Portanto, este trabalho concentra-se na primeira categoria de métodos.

Para remoção dos segmentos de meio-fio, neste trabalho, utilizou-se uma abordagem simples, mas eficiente. Geralmente, o objeto candidato a meio-fio possui uma geometria contínua e horizontalizada. Além disso, em objetos com geometria horizontalizada, a variância das alturas dos pontos pertencentes aos segmentos associados tende a ser menor. Por isso, utilizou-se a média e a variância das alturas dos pontos dos segmentos como critérios para a escolha do segmento candidato a meio-fio.

A média μ das alturas dos pontos de um segmento é calculada pela equação:

$$\mu = \frac{\sum_{i=1}^{N} z_i}{N}, \tag{18}$$

onde z_i é a altura do ponto p_i (Seção 3.2) do segmento e N é o número de pontos do segmento.

A variância σ^2 das alturas z_i dos pontos de um segmento é uma medida de dispersão, que indica "o quão longe" em geral os valores de z_i encontram-se do valor de μ, e é calculada pela equação:

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^{N} (z_i - \mu)^2 \tag{19}$$

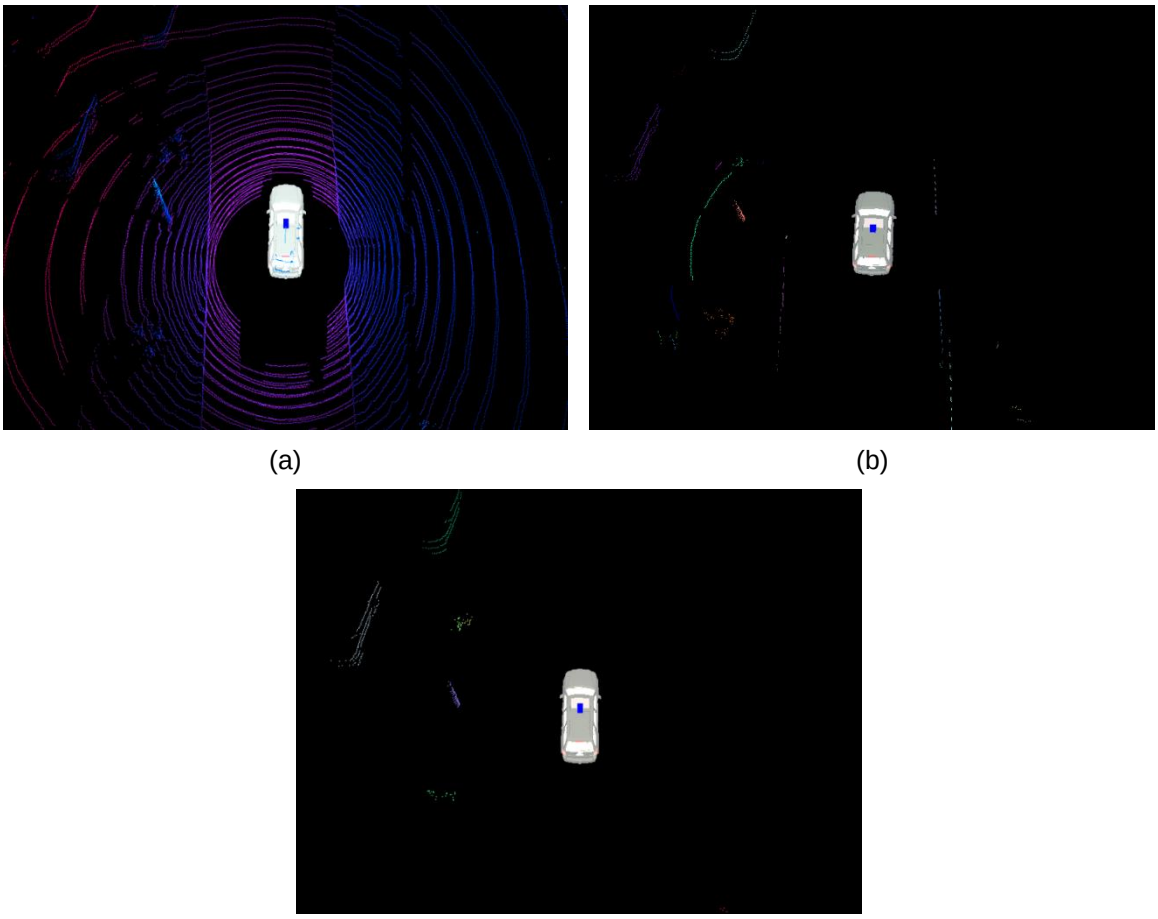
Se μ for inferior a um limiar μ_{limiar} e σ^2 for inferior a um limiar $\sigma^2{}_{limiar}$, este segmento é removido da lista de segmentos identificados na fase de agrupamento dos pontos (Seção 3.2.2). A Figura 22 mostra o pseudocódigo do algoritmo de remoção do segmento de meio-fio.

Entrada: $ListaAtualObjetos = o_1, o_2, \dots, o_M$

1. **para** cada objeto o_j, com $j = 1, \dots, M$, da $ListaAtualObjetos$ **faça**
2. //Calcula média μ das alturas dos pontos do objeto o_j, onde z_i é a altura do ponto do objeto e N é o número de pontos do objeto.
   $\mu = \frac{\sum_{i=1}^{N} z_i}{N}$
3. //Calcula a variância σ^2 das alturas z_i dos pontos do objeto o_j com relação a sua média
   $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(z_i - \mu)^2$
4. **se** $\mu < \mu_{limiar}$ e $\sigma^2 < \sigma^2{}_{limiar}$ **então**
5. Remove o_j da $ListaAtualObjetos$
6. **fim se**
7. **fim para**

Figura 22: Pseudocódigo do algoritmo de remoção do segmento de meio-fio

A Figura 23 mostra um exemplo de remoção do segmento de meio-fio. A Figura 23(a) mostra um exemplo de uma nuvem de pontos, na qual pode ser observado um meio-fio à esquerda e um canteiro central à direita. A Figura 23(b) mostra a nuvem de pontos após sua segmentação e a detecção dos segmentos do meio-fio e do canteiro central; na Figura 23(b), pontos associados a diferentes agrupamentos são destacados em cores diferentes. A Figura 23(c) mostra a nuvem de pontos após a remoção dos segmentos do meio-fio e do canteiro central.



(a) (b)

(c)

Figura 23: Exemplo de remoção do segmento de meio-fio. (a) Exemplo de uma nuvem de pontos 3D, na qual pode ser observado um meio-fio à esquerda e um canteiro central à direita. (b) Nuvem de pontos após sua segmentação e a detecção dos segmentos do meio-fio e do canteiro central; pontos associados a diferentes agrupamentos são destacados em cores diferentes. (c) Nuvem de pontos após a remoção dos segmentos do meio-fio e do canteiro central

3.3 Associação dos Segmentos

Para associação dos segmentos (ou objetos) observados na varredura atual do sensor LIDAR 3D aos mesmos objetos observados em varreduras anteriores, neste trabalho, utilizou-se uma abordagem baseada no centro de massa dos objetos e no algoritmo do vizinho mais próximo. Abordagens semelhantes foram usadas por outros trabalhos da literatura [18] [20] [26] [84].

Inicialmente, para cada objeto $o_j \in ListaAtualObjetos$ (Seção 3.2.2), é obtido o centro de massa $m_j = (x_j, y_j, z_j)$ de sua nuvem de pontos 3D, que passa a representar o_j, e m_j é projetado no plano xy, ou seja, $z_j = 0$. Assim, cada objeto é representado por um simples ponto projetado no plano xy. Em seguida, é definida uma nova lista de objetos, denominada $ListaObjetosAssociados$. Após a primeira varredura do sensor, os objetos da $ListaAtualObjetos$ (Seção 3.2.2) são inseridos na $ListaObjetosAssociados$, juntamente com seus centros de massa. Nas próximas varreduras do sensor, para cada objeto $o_j \in ListaAtualObjetos$, procura-se o vizinho mais próximo $v_i \in ListaObjetosAssociados$ em um círculo com raio $r < a_{limiar}$, onde a_{limiar} é um parâmetro do algoritmo. Como medida de proximidade, utiliza-se a distância Euclidiana entre os centros de massa dos objetos.

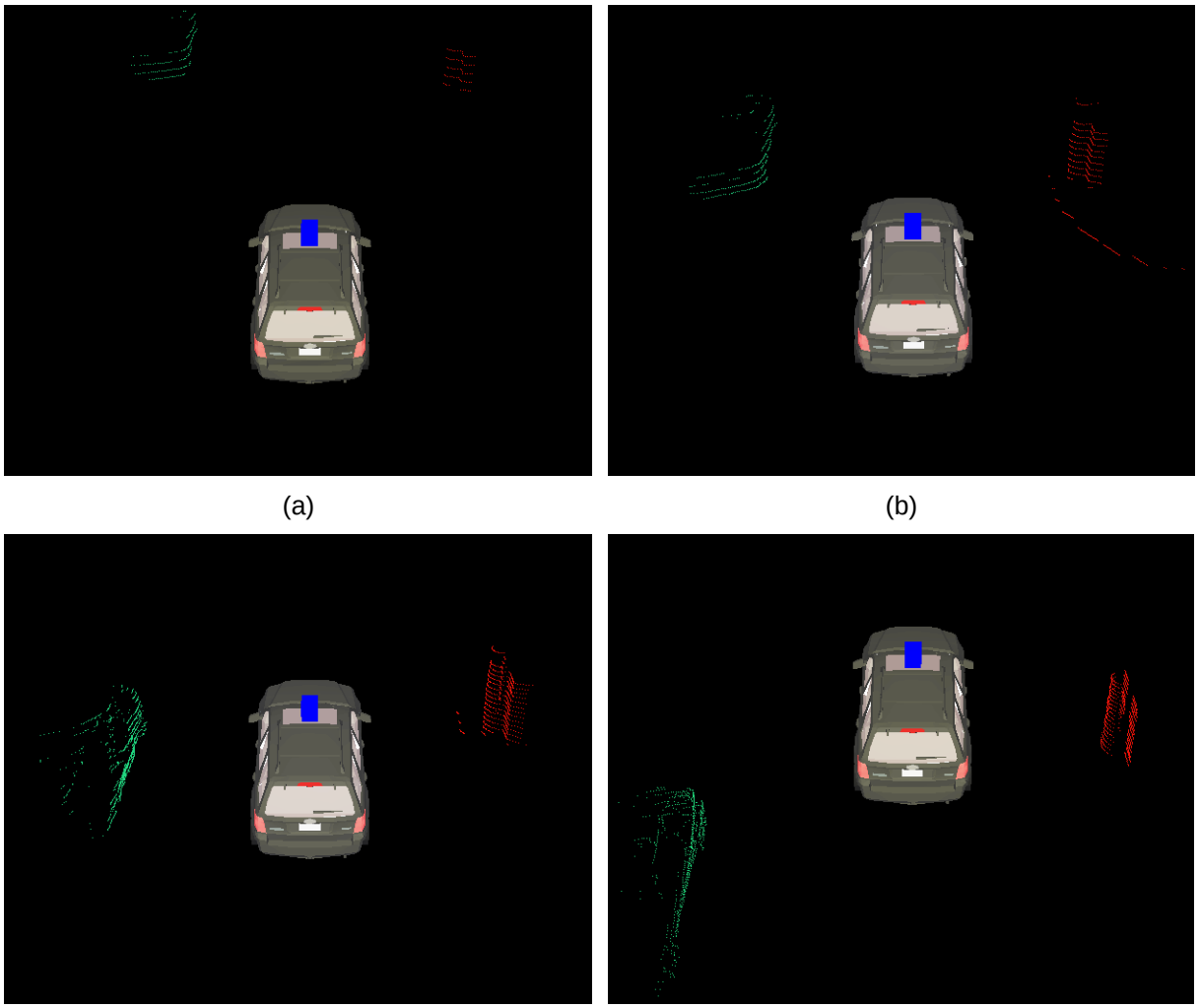
Caso um vizinho mais próximo $v_i \in ListaObjetosAssociados$ a $o_j \in ListaAtualObjetos$ seja encontrado, a nuvem de pontos de v_i é substituída pela nuvem de pontos de o_j e o carimbo de tempo t_{v_i} de v_i é substituído pelo carimbo de tempo t_j de o_j. Além disso, um *flag* é desligado em v_i para indicar que ele já foi associado a um determinado objeto o_j, a fim de evitar que mais de um objeto na $ListaObjetosAssociados$ seja associado a um mesmo objeto na $ListaAtualObjetos$. Caso um vizinho mais próximo $v_i \in ListaObjetosAssociados$ a $o_j \in ListaAtualObjetos$ não seja encontrado, o_j é inserido na $ListaObjetosAssociados$ como um novo objeto observado no ambiente e uma nova cor é atribuída aos pontos deste novo objeto. A Figura 24 mostra o pseudocódigo do algoritmo de associação dos segmentos.

Entrada: $ListaAtualObjetos = o_1, o_2, \dots, o_M$

1. **se** é a primeira varredura do laser **então**
2. $ListaObjetosAssociados = ListaAtualObjetos$
3. **senão**
4. **para** cada objeto o_j, com $j = 1, \dots, M$, da $ListaAtualObjetos$ **faça**
5. Procura-se o vizinho mais próximo $v_i \in ListaObjetosAssociados$ em um círculo com raio r
6. **fim para**
7. //a_{limiar} é um parâmetro do algoritmo
   se $r < a_{limiar}$ **então**
8. // A nuvem de pontos de v_i é substituída pela nuvem de pontos de o_j
   $nuvem_{v_i} = nuvem_{o_j}$
9. // O carimbo de tempo t_{v_i} de v_i é substituído pelo carimbo de tempo t_j de o_j
   $t_{v_i} = t_{o_j}$
   Desliga $Flag_{v_i}$ em v_i para indicar que ele já foi associado a um determinado objeto o_j
10. **senão**
11. // novo objeto observado no ambiente
    Insere o objeto o_j na $ListaObjetosAssociados$
12. **fim se**
13. Liga todos os *Flags* dos objetos $v_i \in ListaObjetosAssociados$
14. **fim se**

Figura 24: Pseudocódigo do algoritmo de associação dos segmentos

A Figura 25 mostra um exemplo de uma sequência de nuvens de pontos após a associação dos segmentos, na qual pode ser observado um veículo se aproximando pela esquerda (representado por pontos verdes) e um obstáculo estático à direita (representado por pontos vermelhos). A Figura 25(a) mostra a nuvem de pontos em um tempo t; a Figura 25(b) em um tempo $t + 1$; a Figura 25(c) em um tempo $t + 2$; e, finalmente, a Figura 25(d) em um tempo $t + 3$.



(a) (b)

(c) (d)

Figura 25: Exemplo de uma sequência de nuvens de pontos após a associação dos segmentos, na qual pode ser observado um veículo se aproximando pela esquerda (representado por pontos verdes) e um obstáculo estático à direita (representado por pontos vermelhos). (a) Nuvem de pontos em um tempo *t*. (b) Nuvem de pontos em um tempo *t* + 1. (c) Nuvem de pontos em um tempo *t* + 2. (d) Nuvem de pontos em um tempo *t* + 3

Os objetos são eliminados da $ListaObjetosAssociados$ quando atingem uma determinada distância Euclidiana do veículo autônomo. Para isso, a cada varredura do sensor, as distâncias entre os centros de massa de cada objeto e a posição global do veiculo autônomo são calculadas e verificadas. As cores usadas pelos objetos eliminados são novamente disponibilizadas para novas associações entre objetos.

3.4 Rastreamento dos Veículos em Movimento

Para rastreamento, i.e., estimativa do estado (posição, orientação e velocidade), dos objetos, neste trabalho, utilizou-se uma variante do filtro de partículas denominada *bootstrap* [74] ou re-amostragem por importância da amostragem (*sampling importance re-sampling* – SIR) [75].

O filtro de partículas representa a função de densidade de probabilidade do estado do objeto por um conjunto de amostras aleatórias (ou partículas), ao invés de por uma função sobre o espaço de estados. A cada iteração, a variante *bootstrap* opera em três fases: predição, correção e re-amostragem [14]. Na fase de predição, um estado é estimado para cada partícula com base no estado anterior da partícula por meio da amostragem de um modelo de transição de estado; o estado é uma amostra de uma distribuição de probabilidade de transição de estados dado o estado anterior. Na fase de correção, um peso é calculado para cada partícula com base em uma medida por meio de um modelo de observação; o peso é a probabilidade da medida sob o estado estimado na fase de predição. Finalmente, na fase de re-amostragem, M partículas são re-amostradas com reposição, onde M é o número de partículas no conjunto de partículas. A probabilidade de amostrar uma partícula é dada pelo seu peso. A re-amostragem transforma um conjunto de M partículas em um outro conjunto de partículas de mesmo tamanho. O novo conjunto usualmente possui muitas duplicatas, porque as partículas são amostradas com reposição. Além disso, as partículas contidas pelo novo conjunto tendem a ser aquelas com maior peso. Maiores detalhes sobre o filtro *bootstrap* podem ser encontrados na Seção 2.3.2.2. Abordagens semelhantes foram usadas por outros trabalhos da literatura [34] [81] [83].

Neste trabalho, um filtro com um conjunto de M partículas é inicializado para cada um dos objetos pertencentes à $ListaObjetosAssociados$ (Seção 3.3). Cada partícula é uma instanciação do estado do objeto no tempo t e é denotada por X_t^m. O estado da partícula é descrito pelo vetor $(x_t^m, y_t^m, \theta_t^m, v_t^m)$, onde (x_t^m, y_t^m) é a posição da partícula, θ_t^m é a orientação e v_t^m é a velocidade. A posição das partículas é inicializada com a posição do centro de massa da nuvem de pontos do objeto (Seção 3.3); a ori-

entação é inicializada com um valor aleatório de uma distribuição uniforme no intervalo $[-\pi,\ \pi]$; e a velocidade é inicializada com um valor aleatório de uma distribuição uniforme no intervalo $[0,25]$ m/s.

O filtro opera quando uma associação de um objeto observado no tempo atual t_j é feita a um objeto observado em um tempo anterior t_k. Na fase de predição, para estimar o estado de cada partícula X_t^m do objeto associado no tempo $t = t_j$ com base no estado da partícula X_{t-1}^m em tempo anterior $t - 1 = t_k$, utilizou-se um modelo de transição de estado baseado no modelo de movimento com velocidade constante, descrito pelas seguintes equações:

$$x_t^m = x_{t-1}^m + x_{t-1}^m * v_{t-1}^m * \Delta t * \cos\theta_{t-1}^m \quad (20)$$

$$y_t^m = y_{t-1}^m + y_{t-1}^m * v_{t-1}^m * \Delta t * \sin\theta_{t-1}^m\ , \quad (21)$$

onde Δt é a diferença entre os carimbos de tempo $t = t_j$ e $t - 1 = t_k$. A cada predição, a orientação e a velocidade são perturbadas por um ruído gaussiano independente, de acordo com as seguintes equações:

$$\theta_t^m = \theta_{t-1}^m + N(\mu, \sigma_1^2) \text{ e} \quad (22)$$

$$v_t^m = v_{t-1}^m + N(\mu, \sigma_2^2), \quad (23)$$

onde as funções $N(\mu, \sigma_1^2)$ e $N(\mu, \sigma_2^2)$ geram amostras aleatórias de uma distribuição normal centrada em zero (i.e., $\mu = 0$) com variâncias σ_1^2 e σ_2^2, respectivamente. A velocidade é mantida no intervalo $[v_{min}, v_{max}] = [0,25]$ m/s. Isso é feito para evitar que ruídos excessivos na velocidade, principalmente causados por objetos (por exemplo, vegetação) movidos rapidamente pelo vento, gerem estados espúrios.

Na fase de correção, utilizou-se um modelo de observação. O intuito deste modelo é gerar, para cada partícula X_t^m, de acordo com a medida do sensor, um peso que representa a probabilidade do estado da partícula ser o estado real do objeto e que é denotado por w_t^m . Este peso é proporcional à distância Euclidiana $dist_t^m =$

$\sqrt{(x_t^m - x_t^j)^2 + (y_t^m - y_t^j)^2}$ entre a posição da partícula X_t^m e o centro de massa do objeto o_t^j, e é calculado pela equação:

$$w_t^m = \exp(-dist_t^m), \tag{24}$$

Note que w_t^m é inversamente proporcional a $dist_t^m$ e que, para valores pequenos de $dist_t^m$, a função $\exp(-dist_t^m)$ fornece valores suficientemente discriminativos. Uma abordagem semelhante foi adotada por K. Al-Mutib [84].

Após a execução da fase de correção, os pesos gerados são normalizados, de acordo com a seguinte equação:

$$w_t^m = \frac{w_t^m}{\sum_{m=1}^{M} w_t^m} \tag{25}$$

Na fase de re-amostragem, um novo conjunto de M partículas é gerado ao re-amostrar com reposição M partículas do conjunto anterior de partículas. A probabilidade de amostrar uma partícula é igual ao seu peso. As partículas do novo conjunto são aquelas com maiores pesos. Sendo assim, a fase de re-amostragem reduz os efeitos do fenômeno da degeneração, pelo qual, depois de algumas iterações do filtro, as partículas recebem pesos insignificantes [75].

Na fase de re-amostragem, utilizou-se um algoritmo de amostragem de baixa variância (*low variance sampling*) semelhante àquele apresentado por S. Thrun et al. [14]. Este algoritmo computa um único número aleatório e seleciona partículas de acordo com este número com uma probabilidade proporcional ao peso da partícula. Inicialmente, o algoritmo cria uma lista de tamanho igual à soma dos pesos das partículas $W = \sum_{m=1}^{M} w_t^m$; cada partícula x_t^m ocupa um trecho da lista igual ao seu peso w_t^m. Em seguida, um número aleatório n é extraído do intervalo $[0, W]$. O algoritmo então seleciona a primeira partícula ao escolher aquela que corresponde a n, e cada uma

das próximas partículas ao adicionar repetidamente o valor W/M a n e escolher aquela que corresponde ao número resultante.

Finalmente, o estado do objeto $\hat{E} = (\hat{x}, \hat{y}, \hat{\theta}, \hat{v})$ é estimado pela média dos estados das partículas $X_t^m = (x_t^m, y_t^m, \theta_t^m, v_t^m)$, $1 \leq m \leq M$, ponderada pelos pesos das partículas [38] [75], segundo a seguinte equação:

$$\hat{E} = \sum_{i=1}^{M} w_t^m \, X_t^m \tag{26}$$

A discriminação entre objetos estáticos e veículos em movimento é feita na fase de rastreamento [25]. Se o estado estimado para o objeto possuir uma velocidade acima de um determinado limiar, o objeto é considerado um veículo em movimento.

Uma caixa delimitadora (*bounding box*) é atribuída ao objeto considerado um veículo em movimento. A caixa delimitadora possui um formato de paralelepípedo retângulo, e seu tamanho é fixo e com medidas semelhantes a de um veículo de passeio. Seu centro está em $(\hat{x}, \hat{y})$ com orientação e velocidade iguais a $\hat{\theta}$ e $\hat{v}$, respectivamente.

A Figura 26 mostra o pseudocódigo do algoritmo de rastreamento utilizando o filtro de partículas *bootstrap* ou re-amostragem por importância da amostragem.

Entrada: Vetor de partículas de tamanho M, onde cada partícula é denotada por $(x_t^m, y_t^m, \theta_t^m, v_t^m)$, centro de massa do objeto (x_t, y_t) e Δt.

1. **se** é a primeira vez do filtro **então** // Inicializa as partículas
2. **para** cada partículaX_t^m, com $m = 1, \dots, M$ **faça**
3. // Posição da partícula de acordo com o centro de massa do objeto $(x_t^m, y_t^m) = (x_t, y_t)$
4. $\theta_t^m = U[-\pi, \pi]$
5. $v_t^m = U[0{,}25]$
6. **fim para**
7. **senão**
8. // Modelo de movimento com velocidade constante
   para cada partículaX_t^m, com $m = 1, \dots, M$ **faça**
9. $x_t^m = x_{t-1}^m + x_{t-1}^m * v_{t-1}^m * \Delta t * cos\theta_{t-1}^m$
10. $y_t^m = y_{t-1}^m + y_{t-1}^m * v_{t-1}^m * \Delta t * sin\theta_{t-1}^m$
11. $\theta_t^m = \theta_{t-1}^m + N(\mu, \sigma_1^2)$
12. $v_t^m = v_{t-1}^m + N(\mu, \sigma_2^2)$
13. **fim para**
14. // Modelo de observação
    para cada partículaX_t^m, com $m = 1, \dots, M$ **faça**
15. // Calcula a distância Euclidiana
    $dist_t^m = \sqrt{(x_t^m - x_t^j)^2 + (y_t^m - y_t^j)^2}$
    $w_t^m = exp(-dist_t^m)$ // Computa o peso da partícula
16. **fim para**
17. **para** cada partícula X_t^m, com $m = 1, \dots, M$ **faça**
18. $w_t^m = \frac{w_t^m}{\sum_{m=1}^M w_t^m}$ // Normaliza os pesos
19. **fim para**
20. // Re-amostragem das partículas.
    Replica as partículas na proporção de seus pesos
21. Computa o estado do objeto $\hat{E} = (\hat{x}, \hat{y}, \hat{\theta}, \hat{v})$ pela média dos estados das partículas ponderada pelos pesos das partículas: $\hat{E} = \sum_{i=1}^M w_t^m X_t^m$
22. **fim se**

Figura 26: Pseudocódigo do algoritmo de rastreamento utilizando o filtro de partículas *bootstrap* ou re-amostragem por importância da amostragem

4 MATERIAIS E METODOLOGIA EXPERIMENTAL

Este capítulo apresenta a metodologia e os materiais utilizados neste trabalho. A seção 4.1 descreve a plataforma robótica IARA e os seus componentes. A Seção 4.2 apresenta o sensor LIDAR 3D. A Seção 4.3 introduz o *framework* CARMEN, responsável pela comunicação entre os módulos de *software* e a Seção 4.4 cita a biblioteca utilizada no desenvolvimento do DATMO. A Seção 4.5 apresenta os detalhes da implementação do DATMO e sua integração ao *framework* CARMEN. Por fim, a Seção 4.6 descreve como as bases de dados utilizadas nos experimentos foram obtidas.

4.1 *Intelligent Autonomous Robotic Automobile* (IARA)

Para testar os algoritmos desenvolvidos neste trabalho, foram utilizados os dados gerados pelos sensores adaptados ao veículo robótico autônomo IARA (*Intelligent Autonomous Robotic Automobile*) IARA foi construída com base no automóvel de passeio *Ford Escape Hybrid* (Figura 27). Esse automóvel passou por diversas adaptações, como a instalação de sensores e a instalação de mecanismos de controle do acelerador, freio e posição do volante por meio dos computadores instalados no porta-malas (Figura 28).

Figura 27: Veículo robótico autônomo IARA

A tecnologia eletrônica de acionamento dos atuadores (volante, acelerador, freio, entre outros) do automóvel foi desenvolvida pela empresa *Torc Robotics* (Virginia, USA) [88].

Figura 28: Recursos computacionais do veículo robótico autônomo IARA [86]

Os sensores disponíveis no veículo robótico incluem câmeras estéreo *Bumblebee XB3* da *Point Grey* (Colúmbia Britânica, Canadá) (http://ww2.ptgrey.com/stereo-vision/bumblebee-xb3), um *Light Detection And Ranging* (LIDAR) HDL-32E da *Velodyne* (Califórnia, USA) (http://velodynelidar.com/lidar/hdlproducts/hdl32e.aspx), e um *Attitude and Heading Reference System* (AHRS) MTi da *Xsens* (Califórnia, USA) (http://www.xsens.com/products/mti/). A Figura 29 demonstra como os sensores foram instalados no veículo robótico autônomo IARA.

Figura 29: Sensores do veículo robótico autônomo IARA [86]

4.2 Sensor LIDAR 3D

Neste trabalho vamos nos concentrar no *Light Detection And Ranging* (LIDAR) HDL-32E da *Velodyne* para sensoriar o ambiente (Figura 30). Sensores Velodyne HDL-32E fornecem uma varredura em 3D do ambiente em altas taxas de quadros (10 Hz) e produzem nuvens de aproximadamente 700.000 pontos por segundo.

Figura 30: LIDAR HDL-32E da *Velodyne*

Utilizam comunicação *ethernet* através do protocolo UDP e possuem a capacidade de sincronizar *timesamp* com o GPS. Veja Figura 31.

Figura 31: Conexões de dados do LIDAR HDL-32E

O *Velodyne* possui um raio de atuação de até 70 metros e é capaz de medir distâncias com um erro médio inferior a 2 centímetros a 25 metros. A resolução angular vertical é de aproximadamente 1.33 graus e a horizontal de aproximadamente 0.16 graus. Possui um campo visual vertical entre +10.67 e -30.67 graus [89].

4.3 *Carnegie Mellon* (CARMEN) *Robot Navigation Toolkit*

Os algoritmos implementados neste trabalho foram desenvolvidos na linguagem C/C++ como um módulo do *Carnegie Mellon Robot Navigation Toolkit* (*framework* CARMEN). CARMEN é um conjunto de softwares para controle de robôs. Ele foi projetado para fornecer uma interface consistente e um conjunto de primitivas básicas para pesquisa em robótica em uma ampla variedade de plataformas de robôs comerciais. O objetivo do framework é diminuir a barreira entre o desenvolvimento de novos códigos tanto para robôs simulados quanto para robôs reais, e também facilitar o compartilhamento de algoritmos entre diferentes instituições [90].

CARMEN utiliza uma arquitetura composta por diversos softwares modulares, onde cada módulo representa uma habilidade do robô independente das demais. Entre os módulos existentes no CARMEN estão: o módulo de localização, o módulo de plane-

jamento de movimento e o módulo de detecção e rastreamento de veículos em movimento. Os módulos se comunicam uns com os outros através de um protocolo de comunicação chamado IPC (*Inter Process Communication*), desenvolvido por Reid Simmons em 1994 para apoiar um projeto da NASA [91].

Um atrativo do IPC é a possibilidade de comunicação entre processos sendo executados em diferentes computadores por meio do protocolo TCP/IP. Além disso, uma grande variedade de linguagens possui biblioteca IPC e ele é suportado por vários sistemas operacionais.

IPC utiliza principalmente o paradigma de comunicação *Publish-Subscribe*. Neste modelo, a gerência da comunicação é realizada por um processo chamado Central, que é responsável por receber as mensagens publicadas e repassá-las para os módulos interessados. Os módulos que publicam mensagens são chamados de *Publishers*, enquanto que os módulos que recebem mensagens são chamados de *Subscribers*. Um módulo pode desempenhar um papel de *Publisher* e de *Subscriber* simultaneamente. Um módulo se torna um *Subscriber* assim que ele demonstra interesse em receber uma determinada mensagem, representado por uma assinatura da mensagem enviada ao Central. Já para se tornar um *Publisher*, um módulo precisa publicar uma mensagem. O envio de uma nova mensagem acontece através de um pedido de publicação enviado ao Central [86].

Algumas vantagens na utilização do *framework* CARMEN juntamente com a comunicação pelo IPC podem ser citadas como [90]:

- Flexibilidade: um usuário pode combinar diferentes bases de robôs e sensores, simplesmente executando os módulos apropriados. Isto elimina a necessidade de acomodar todas as configurações possíveis de hardware em um único processo.
- Escalabilidade: os módulos não precisam ser executados em um mesmo processador. Módulos críticos podem ser executados em um computador separado ou até mesmo fora do robô e a comunicação acontece pela rede.

- Confiabilidade: os outros módulos não param de funcionar se um ou mais módulos falharem. A única exceção é o módulo Central.

O desenvolvimento da versão *open source* do CARMEN foi descontinuado em 2008. Contudo, este *framework* garantiu a seus desenvolvedores a vitória na *Defense Advanced Research Projects Agency* (DARPA) *Grand Challege* de 2005 [27] e o segundo lugar na DARPA *Urban Challenge* [29]. Assim, embora existam outros *frameworks open source* hoje disponíveis e mantidos, a equipe do LCAD optou por continuar o desenvolvimento do CARMEN no LCAD [86].

Mais informações sobre o LCAD-CARMEN, estão disponíveis no link http://www.lcad.inf.ufes.br/wiki/index.php/Carmen_Robot_Navigation_Toolkit.

4.4 *Point Cloud Library* (PCL)

A Biblioteca PCL (*Point Cloud Library*) é resultado de um projeto de código aberto referente a processamento de nuvens de pontos. Ela recebe suporte da comunidade internacional de pesquisa e de desenvolvedores tanto na área de robótica quanto de percepção artificial. A PCL possui diversos algoritmos, incluindo filtros, estimativa de *features*, reconstrução de superfícies, registro e segmentação de nuvens de pontos que podem ser utilizados em projetos, como por exemplo, filtrar os *outliers* a partir de dados ruidosos, segmentar partes relevantes de uma cena, e etc, utilizando as linguagens C/C++ [53]. Neste trabalho, foi utilizada a implementação do algoritmo *Euclidean Cluster Extraction* fornecido por essa biblioteca na versão 1.7 (http://pointclouds.org) através de sua compilação para Linux, além de diversas outros métodos disponíveis, entre eles: método para obtenção do centroide, autovalores e autovetores, pontos máximo e mínimo, matriz de covariância, entre outros.

4.5 Detalhes de Implementação do Sistema de Detecção e Rastreamento de Veículos em Movimento

A Figura 32 apresenta um diagrama de fluxo de dados ilustrando os programas que compõem o sistema de DATMO e os dados transmitidos entre estes programas. O módulo do sensor (hexágonos na cor salmão) gera dados para os demais módulos. Os módulos de processamento (caixas na cor verde) recebem dados e geram novos dados.

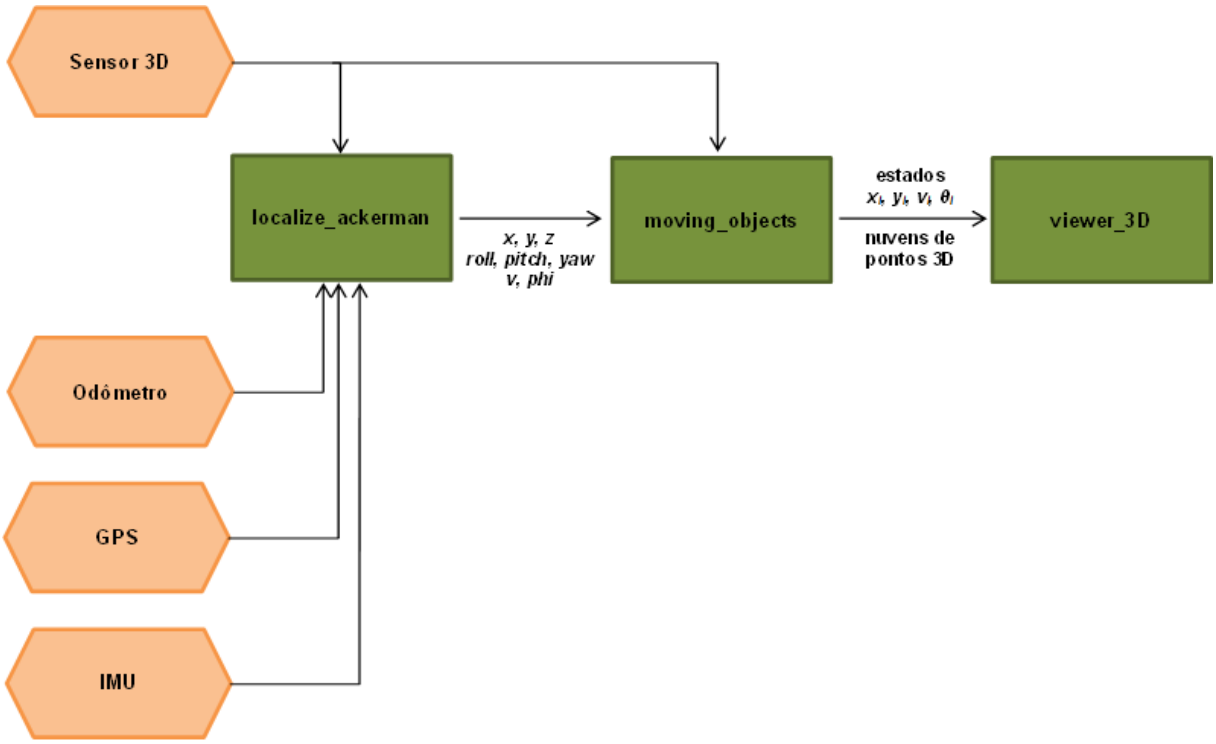


Figura 32: Diagrama de fluxo de dados ilustrando os programas que compõem o sistema de DATMO e os dados transmitidos entre estes programas

Os módulos do CARMEN que compõe sistema de DATMO são listados abaixo:

- *localize_ackerman*: este módulo recebe como entrada as mensagens dos sensores (odometria, *Velodyne*, GPS e IMU) e realiza a otimização das poses do veículo autônomo robótico e as publica na forma de uma mensagem inte-

grada contendo uma pose 6D (posição 3D, x, y, z e orientação 3D, *roll*, *pitch* e *yaw*), ângulo do volante e a velocidade do veiculo.

- *moving_objects*: este módulo recebe como entrada a mensagem do sensor *Velodyne* e uma mensagem contendo uma pose 6D (posição 3D, x, y, z e orientação 3D, *roll*, *pitch* e *yaw*), ângulo do volante e a velocidade do veículo. Publica na forma de uma mensagem integrada contendo um conjunto nuvens de pontos 3D (*x*, *y*, *z*) representando os objetos em cena e seus os estados estimados (x, y, velocidade e orientação).
- *viewer_3D*: este módulo recebe como entrada as nuvens de pontos 3D (*x*, *y*, *z*) representando os objetos em cena e seus estados estimados (x, y, velocidade orientação) e gera como saída, em um ambiente visual 3D, caixas delimitadoras 3D (*bounding box*) de tamanho fixo e com medidas semelhantes a de um veículo de passeio sobre cada objeto rastreado. Um número sobre a caixa delimitadora é apresentado e está associado à identificação do rastreamento.

No módulo *moving_objects* várias funções foram criadas no desenvolvimento do sistema de DATMO, sendo as mais significativas descritas abaixo:

- *build_point_cloud_using_velodyne_message*: essa função recebe como entrada dados do sensor, parâmetros do sensor, como posição no veículo e dados de localização do veículo robótico autônomo e gera como saída uma nuvem de pontos com coordenadas globais sem os pontos pertencentes ao solo e sem os pontos acima de um limiar, que neste trabalho foi fixado em 2 metros.
- *pcl_euclidean_cluster_extraction*: essa função pertence a biblioteca da PCL e recebe como entrada uma nuvem de pontos. Sua saída é uma lista de agrupamentos 3D. Nessa função dois parâmetros foram utilizados:
    - *K*: Número mínimo de pontos que um agrupamento precisa conter para ser considerado válido.
    - *D*: A tolerância espacial entre os pontos referente ao raio de uma esfera.

Neste trabalho, *K* e *D* assumem 15 e 0.5 metros, respectivamente.

- *filter_curbs_point_cloud:* essa função recebe como entrada uma lista de agrupamentos 3D e gera uma nova lista de agrupamentos 3D com os meios-fios removidos. A média da altura dos pontos e a variância das alturas com relação à média são utilizadas como parâmetros. Seus valores foram 0.5 metros e 0.02 metros, respectivamente.
- *associate_object_point_clouds_with_previous_point_clouds*: essa função recebe como entrada uma lista de agrupamentos 3D (objetos) e gera uma nova lista com objetos associados e não associados. Utiliza como parâmetro a distância Euclidiana entre os centros de massa dos objetos. Neste trabalho esse valor foi de 2.0 metros.
- *particle_filter_moving_objects_tracking*: essa função recebe como entrada uma lista de objetos, atualiza os filtros de partículas dos objetos associados e gera novos filtros para objetos não associados e disponibiliza os estados *a posteriori* de cada objeto. Os objetos com velocidade acima de um determinado limiar são considerados veículos em movimento. O limiar considerado neste trabalho foi de 3.0 m/s.

A Tabela 1 sumariza os parâmetros utilizados pelo conjunto de algoritmos nas etapas de segmentação e associação de dados.

Tabela 1: Parâmetros utilizados nos algoritmos das etapas de segmentação e associação de dados do sistema de DATMO

Parâmetro	**Valor**
Altura máximo da nuvem de pontos 3D avaliada pelos algoritmos (metros)	2.0
Mínimo de pontos para um agrupamento 3D (pontos 3D)	15
Tolerância espacial entre os pontos para um agrupamento 3D (metros)	0.5
Média da altura dos pontos para remoção do meio-fio (metros)	0.5
Variância das alturas com relação à média para remoção do meio-fio (metros)	0.02
Limiar de distância dos centros de massa para associação dos objetos (metros)	2.0

A Tabela 2 traz uma sumarização dos parâmetros utilizados na etapa de rastreamento através do filtro de partículas. Em todos os experimentos foram utilizados os mesmos parâmetros.

Tabela 2: Parâmetros utilizados no sistema de DATMO na etapa de rastreamento através do filtro de partículas.

Parâmetro	Valor
Número de Partículas	500
Velocidade inicial para rastreamento (m/s)	3.0
Intervalo do valor da velocidade na inicialização aleatória das partículas (m/s)	[0,25]
Intervalo do valor da orientação na inicialização aleatória das partículas (rad)	[-π, π]
Variância da velocidade na propagação das partículas (m/s)	1.0
Variância da orientação na propagação das partículas (rad)	0.25

4.6 Base de Dados

Esta Seção descreve como a base de dados utilizada neste trabalho foi obtida.

O desempenho do sistema de DATMO proposto foi avaliado usando dados de um sensor LIDAR 3D, além de dados de outros sensores, coletados por um veículo autônomo ao longo de uma volta pelo anel viário do campus da UFES. Estes dados foram armazenados em um arquivo de *log*. Durante a execução dos experimentos, um módulo reproduziu este arquivo de *log*, enviando os dados dos sensores nele armazenados para os módulos de interesse.

Para gerar o arquivo de *log*, um motorista humano guiou IARA pelo anel viário do campus da UFES. O trajeto percorrido foi de aproximadamente 3.7 km. O arquivo de *log* foi gravado no dia 20/03/2014 e contém 11 minutos de dados do laser. A Figura 33 traz uma ilustração do percurso realizado.

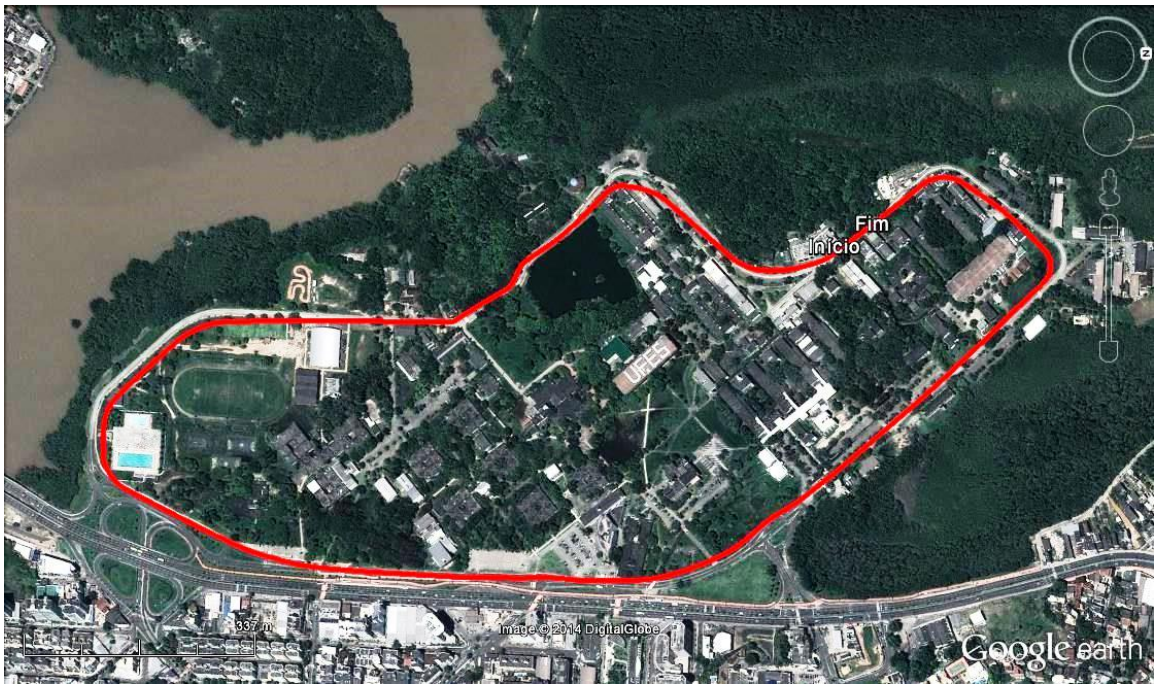


Figura 33: Percurso realizado no anel viário da UFES

5 RESULTADOS EXPERIMENTAIS

Este capítulo demonstra e discute os resultados experimentais obtidos.

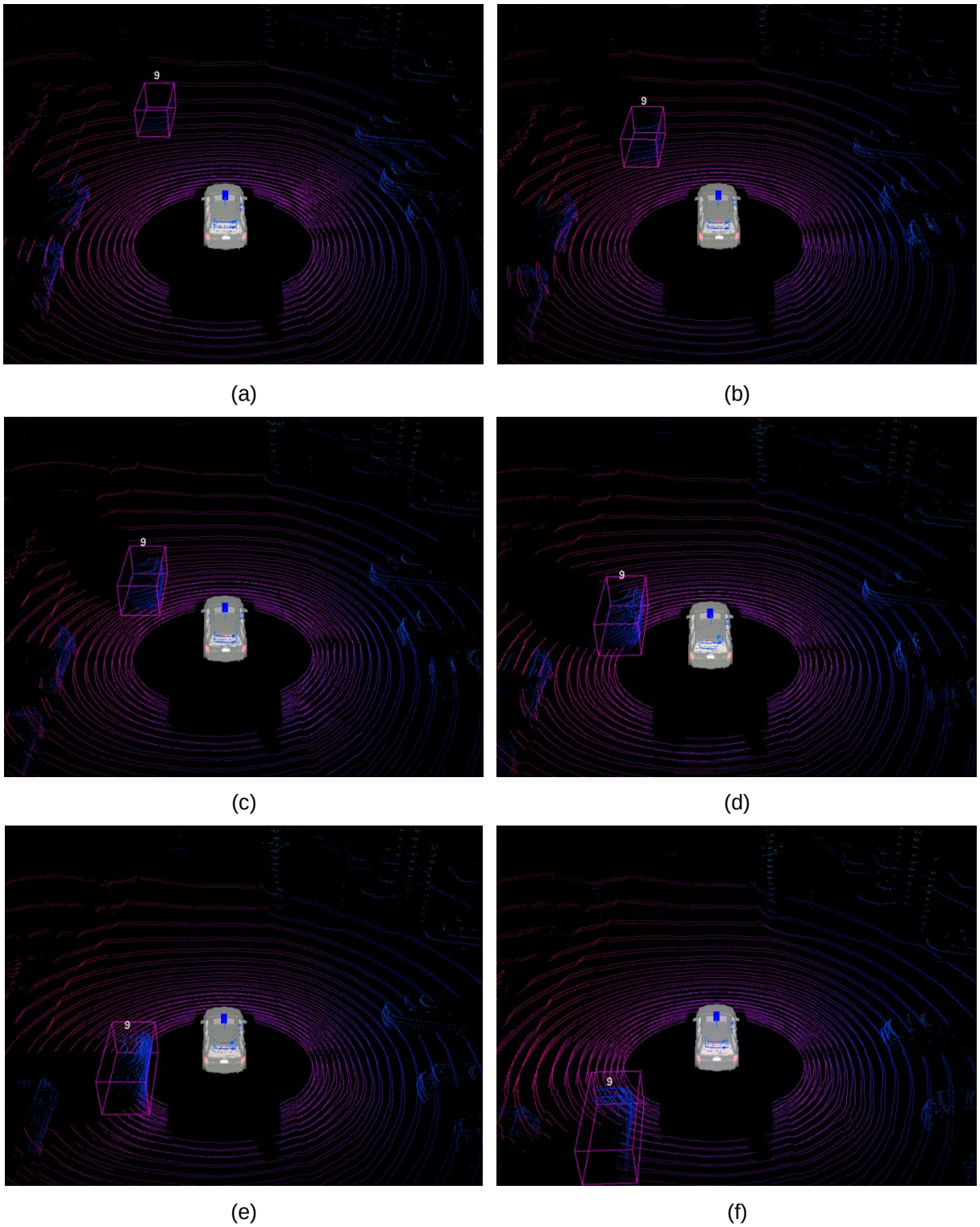
5.1 Resultados Qualitativos

O desempenho do sistema de DATMO proposto foi avaliado qualitativamente por meio da comparação visual entre a nuvem de pontos 3D com coordenadas globais gerada a partir dos dados do sensor LIDAR 3D (Seção 3.2) e a saída do sistema de DATMO, i.e., caixas delimitadoras atribuídas aos objetos considerados veículos em movimento (Seção 3.4). Para melhor visualização, a saída do sistema de DATMO é sobreposta à nuvem de pontos.

A Figura 34 mostra um exemplo de rastreamento de um carro se aproximando pela esquerda do IARA. Na Figura 34, os números em branco sobre as caixas delimitadoras em rosa identificam o veículo que está sendo rastreado. A Figura 34(a) mostra o rastreamento em um tempo t; a Figura 34(b) em um tempo $t + 1$; a Figura 34(c) em um tempo $t + 2$; a Figura 34(d) em um tempo $t + 3$; a Figura 34(e) em um po $t + 4$; e, finalmente, a Figura 34(f) em um tempo $t + 5$. Nesta figura e nas que se seguem, o intervalo entre tempos consecutivos $(t, t + 1, \ldots, t + 5)$ foi determinado manualmente ao manipular a ferramenta do Carmen para reprodução do arquivo de *log* que armazena os dados dos sensores (Seção 4.6).

A Figura 35 mostra um exemplo de rastreamento de dois carros que avançam pela esquerda em direção contrária ao IARA.

A Figura 36 mostra um exemplo de rastreamento de três carros que passam à esquerda do IARA pelo outro lado do canteiro central.



(a) (b)

(c) (d)

(e) (f)

Figura 34: Exemplo de rastreamento de um carro se aproximando pela esquerda do IARA. Os números em branco sobre as caixas delimitadoras em rosa identificam o veículo que está sendo rastreado. (a) Rastreamento em um tempo t. (b) Rastreamento em um tempo$t + 1$. (c) Rastreamento em um tempo $t + 2$. (d) Rastreamento em um tempo $t + 3$. (e) Rastreamento em um tempo $t + 4$. (f) Rastreamento em um tempo $t + 5$

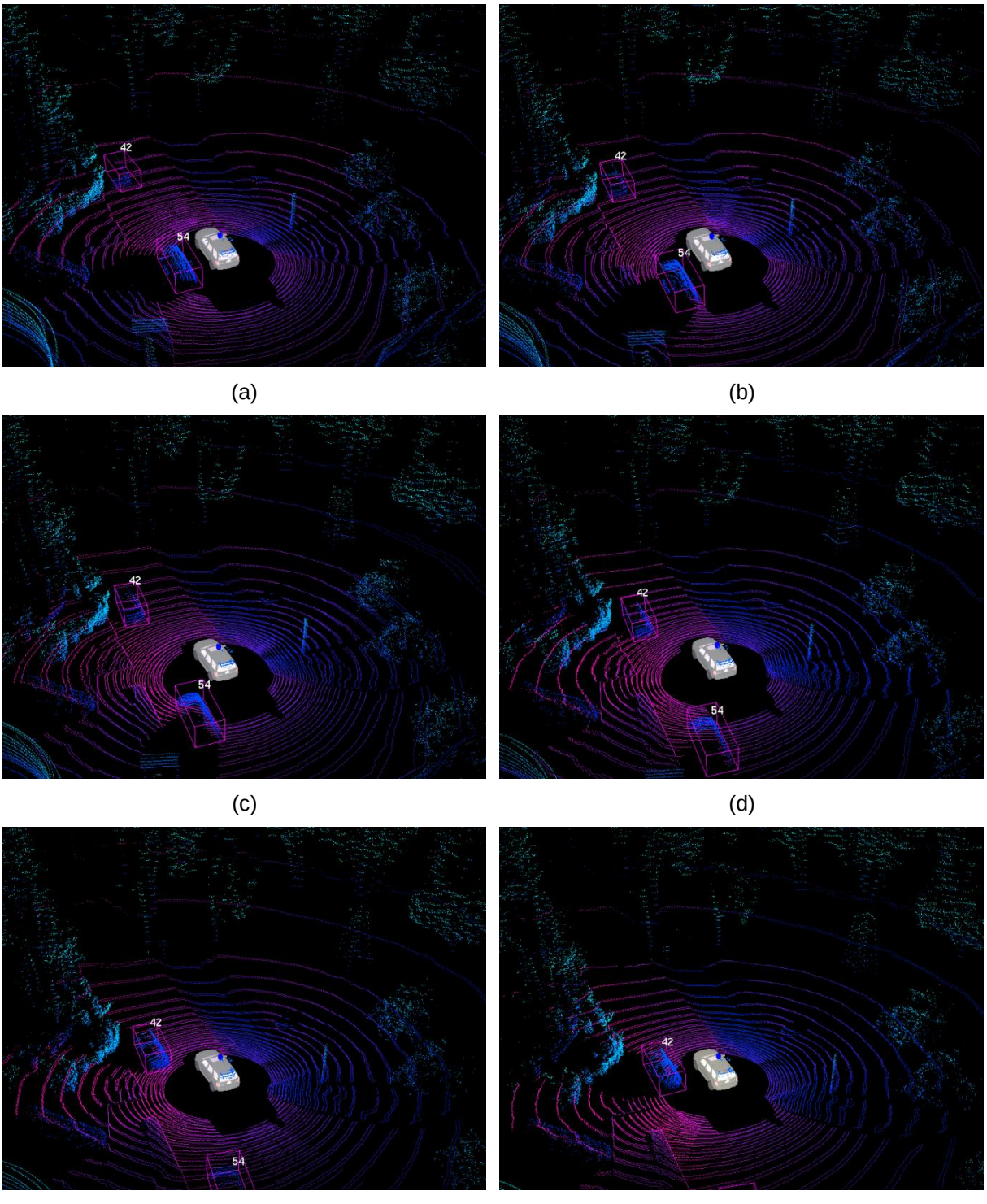


Figura 35: Exemplo de rastreamento de dois carros que avançam pela esquerda em direção contrária ao IARA. Os números em branco sobre as caixas delimitadoras em rosa identificam os veículos que estão sendo rastreados. (a) Rastreamento em um tempo t. (b) Rastreamento em um tempo$t + 1$. (c) Rastreamento em um tempo $t + 2$. (d) Rastreamento em um tempo $t + 3$. (e) Rastreamento em um tempo $t + 4$. (f) Rastreamento em um tempo $t + 5$

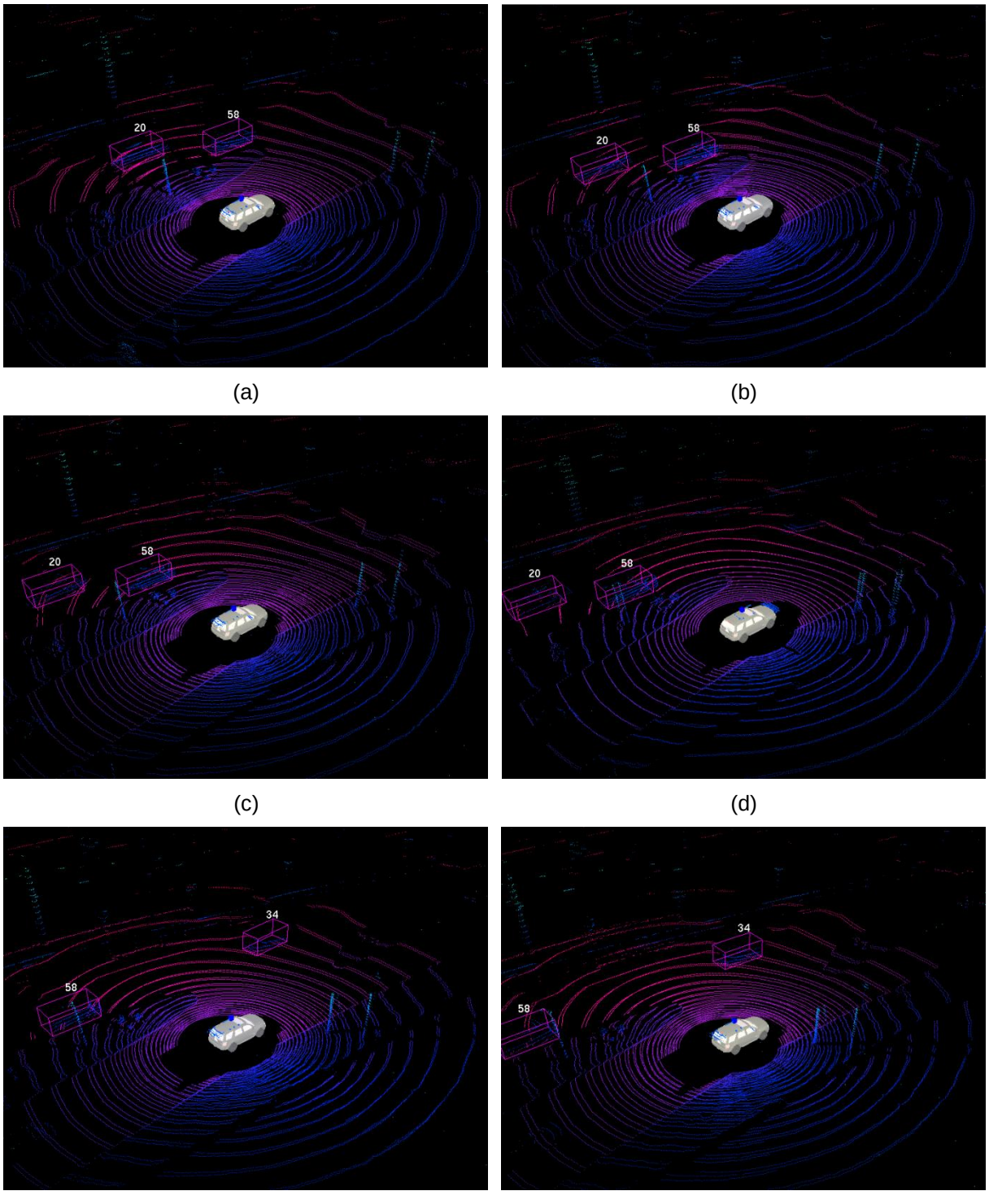
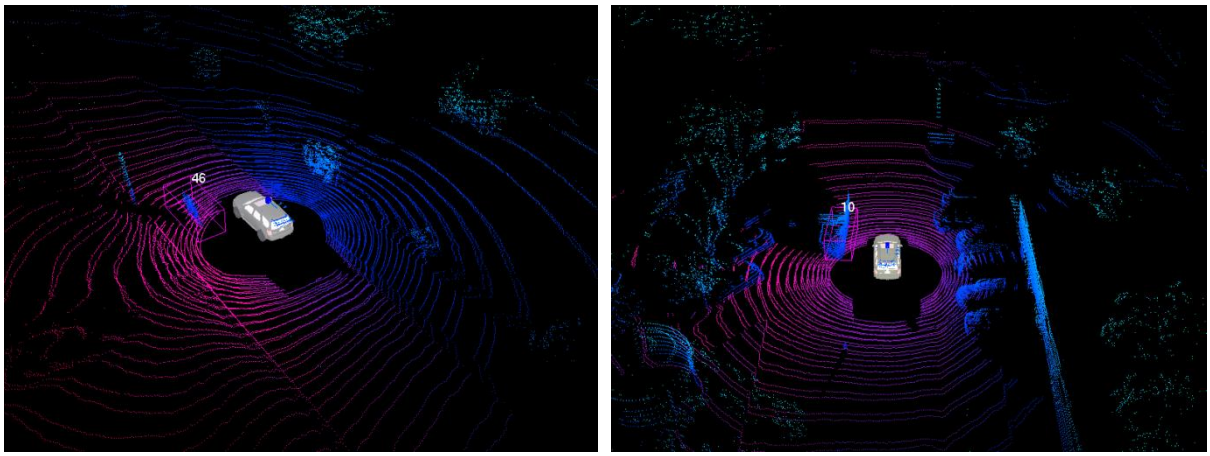


Figura 36: Exemplo de rastreamento de três carros que passam à esquerda do IARA pelo outro lado do canteiro central. Os números em branco sobre as caixas delimitadoras em rosa identificam os veículos que estão sendo rastreados. (a) Rastreamento em um tempo t. (b) Rastreamento em um po$t + 1$. (c) Rastreamento em um tempo $t + 2$. (d) Rastreamento em um tempo $t + 3$. (e) Rastreamento em um tempo $t + 4$. (f) Rastreamento em um tempo $t + 5$

A Figura 37 mostra um exemplo de detecção de uma moto e de um caminhão. Na Figura 37(a), uma moto é detectada e, na Figura 37(b), um caminhão é identificado.



(a) (b)

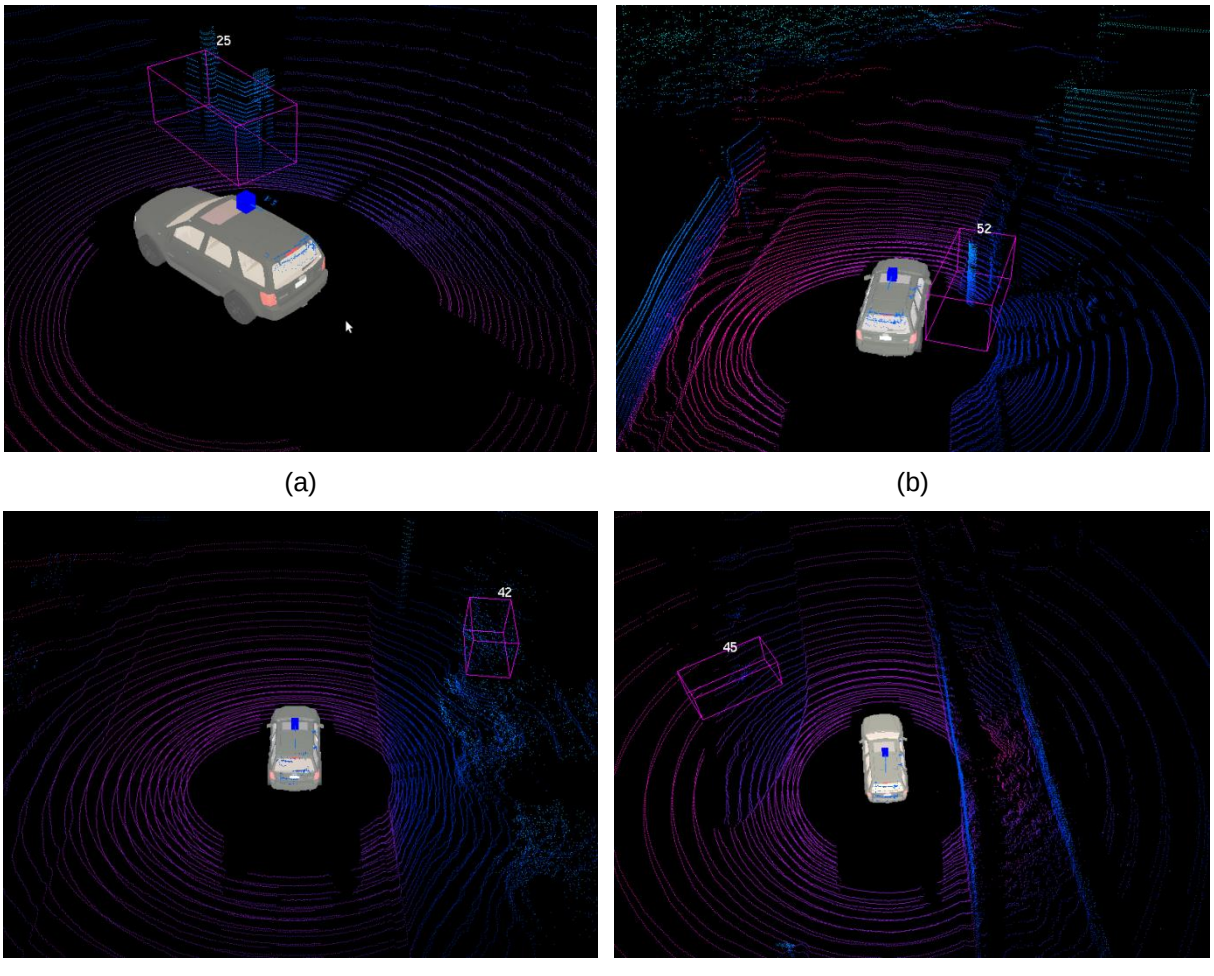
Figura 37: Exemplo de detecção de uma moto e de um caminhão. Os números em branco sobre as caixas delimitadoras em rosa identificam os veículos que estão sendo rastreados. (a) Exemplo de detecção de uma moto. (b) Exemplo de detecção de um caminhão.

Como observado nas figuras mostradas acima, o sistema de DATMO proposto foi capaz de detectar e rastrear objetos com velocidade acima de 3.0 m/s. Entretanto, em alguns momentos, o sistema de DATMO detectou falsos positivos. Três causas desta falha são: mudança do ângulo de observação dos objetos estáticos, a qual é provocada pela movimentação do sensor LIDAR 3D instalado no IARA; proximidade dos objetos estáticos observados; e movimentação rápida pelo vento de componentes dos objetos estáticos observados. Por vezes, quando o objeto estático observado apresenta contornos sinuosos, à medida que IARA se desloca, o centro de massa do objeto pode se deslocar abruptamente. Nestes cenários, a causa da falha é a mudança do ângulo de observação do objeto.

Por outras vezes, na fase de segmentação, pode ocorrer o agrupamento de objetos estáticos diferentes; à medida que IARA se desloca, o centro de massa dos objetos agrupados pode se deslocar abruptamente. Nestes outros cenários, a causa da falha é tanto a mudança do ângulo de observação dos objetos quanto a proximidade dos objetos observados. Finalmente, por outras vezes, quando o objeto estático observado apresenta componentes leves, pode ocorrer a movimentação rápida pelo vento

dos componentes deste objeto; o centro de massa do objeto pode se deslocar abruptamente.

A Figura 38 mostra exemplos de detecções de falsos positivos. A Figura 38(a) mostra um exemplo de detecção de um objeto com contornos sinuosos. A Figura 38(b) mostra um exemplo de detecção de um poste próximo ao meio-fio. A Figura 38(c) mostra um exemplo de detecção de folhas de uma árvore. A Figura 38(d) mostra a detecção de um arbusto no canteiro central.



(a) (b)

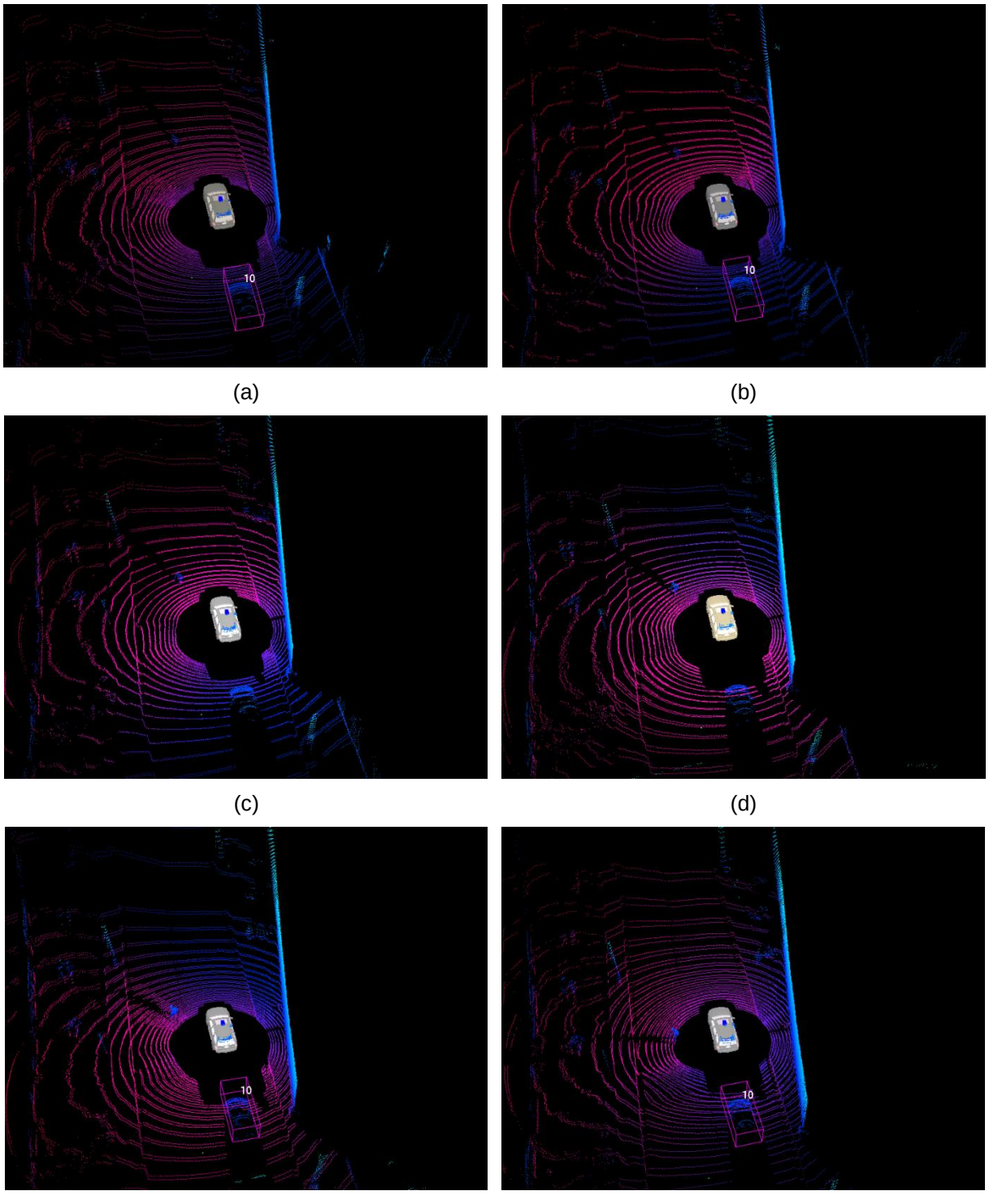
(c) (d)

Figura 38: Exemplos de detecções de falsos positivos. (a) Exemplo de detecção de um objeto com contornos sinuosos. (b) Exemplo de detecção de um poste próximo ao meio-fio. (c) Exemplo de detecção de folhas de uma árvore. (d) Exemplo de detecção de detecção de um arbusto no canteiro central

Apesar do sistema de DATMO proposto detectar falsos positivos, o seu filtro de partículas foi eficiente na discriminação entre objetos estáticos e objetos em movimento ao longo do tempo. Quando a velocidade estimada para um objeto estático está acima do limiar de velocidade usado para definir se o objeto é um veículo em movimento, o filtro rapidamente corrige esta estimativa de velocidade. Na maioria dos casos, poucas iterações do filtro foram necessárias para corrigir a velocidade estimada para um objeto estático.

Em alguns momentos, o sistema de DATMO proposto descontinuou o rastreamento. A caixa delimitadora deixou de ser apresentada em alguns instantes, mas o filtro continuou em processo de propagação. Isto foi observado principalmente no rastreamento de veículos situados atrás do IARA. A causa desta falha seria a redução da velocidade do IARA para uma velocidade abaixo do limiar de velocidade definido. Isto provocaria a redução da velocidade do veículo que está sendo rastreado e que está situado atrás do IARA.

A Figura 39 mostra um exemplo de descontinuidade no rastreamento de um carro.



(a) (b)

(c) (d)

(e) (f)

Figura 39: Exemplo de descontinuidade no rastreamento de um carro. Os números em branco sobre as caixas delimitadoras em rosa identificam o veículo que está sendo rastreado. (a) Rastreamento em um tempo t. (b) Rastreamento em um tempo$t + 1$. (c) Rastreamento em um tempo $t + 2$. (d) Rastreamento em um tempo $t + 3$. (e) Rastreamento em um tempo $t + 4$. (f) Rastreamento em um tempo $t + 5$

5.2 Resultados Quantitativos

Para avaliar o sistema de DATMO proposto de forma quantitativa, foram usados dois índices de avaliação: a precisão ($prec$) e a revocação (rev). A taxa de precisão expressa a relação entre a quantidade de objetos corretamente classificados como veículos em movimento e a quantidade de objetos classificados como veículos em movimento. A taxa de revocação expressa a relação entre a quantidade de objetos corretamente classificados como veículos em movimento pelo sistema e a quantidade real de objetos correspondentes a veículos em movimento. Ambos os valores de precisão e revocação variam entre 0 e 1. Quanto mais próximo de 1, melhor a qualidade da classificação.

As variáveis a serem disponibilizadas para as análises dos dados através da precisão e da revocação são: (i) número de verdadeiros positivos, (ii) número de falsos positivos e (iii) número de falsos negativos. Um verdadeiro positivo (*VP*) ocorre quando o sistema proposto identifica um objeto de forma correta como veículo em movimento. Um falso positivo (*FP*) ocorre quando o objeto detectado não é um veículo em movimento. Um falso negativo (*FN*) ocorre quando um veículo em movimento não é detectado pelo sistema proposto. A partir dessas variáveis são calculadas as medidas de precisão e revocação, de acordo com as seguintes equações:

$$prec = \frac{(VP)}{(VP + FP)} \tag{27}$$

$$rev = \frac{(VP)}{(VP + FN)} \tag{28}$$

O processo de contabilizar os verdadeiros positivos, falsos positivos e falsos negativos, valores necessários para o cálculo das medidas $prec$ e rev, foi executado manualmente pela análise das nuvens de pontos 3D com coordenadas globais geradas a partir dos dados do sensor LIDAR 3D e a saída do sistema de DATMO proposto.

Neste trabalho, um falso positivo é contabilizado todas as vezes que o sistema proposto identifica um objeto que não é um veículo em movimento durante um número consecutivo de *frames* (2, 4, 6, 8 ou 10 *frames*). Um falso negativo é contabilizado quando um veículo em movimento não é detectado ou quando ocorre uma descontinuidade na detecção durante um número consecutivo de *frames*. O número de verdadeiros positivos é determinado pela diferença entre o número total de veículos em movimento (*T*) e os falsos negativos (i.e., $VP = T - FN$), de acordo com o número consecutivo de *frames*. Por exemplo, suponhamos que um poste tenha sido detectado como um veículo em movimento durante 3 *frames* consecutivos, mas identificado como um objeto estático a partir do quarto *frame*. Então, ele será considerado um falso positivo na contabilização associada a 2 *frames* consecutivos e um verdadeiro negativo na contabilização associada a 4 ou mais *frames* consecutivos.

Nos experimentos foram considerados os dados do *frame* com carimbo de tempo 1395343603,399089 segundos até o *frame* com carimbo de tempo 1395344171,828476 segundos, totalizado 9 minutos e 28 segundos. Foram considerados ainda todos os veículos no campo de percepção do sensor, que estavam no anel viário da UFES e cujos pontos 3D foram agrupados em um segmento. Os veículos que saíram do campo de percepção do sensor e após algum tempo voltaram a ser sensoriados são contabilizados como um novo veículo. A Tabela 3 mostra a quantidade de veículos em movimento, descriminada por tipo, presentes nos dados analisados.

Tabela 3: Quantidade de veículos em movimento, descriminada por tipo, presentes nos dados analisados

Tipo de Veículo em Movimento	**Quantidade**
Carro	45
Moto	02
Caminhão	02
Total	49

A Tabela 4 demonstra os valores obtidos de *FP* (coluna 2), *FN* (coluna 3) e *VP* (coluna 4) para um número consecutivo de *frames* (coluna 1).

Tabela 4: Valores obtidos de FP, FN e VP para 2, 4, 6, 8 e 10 *frames*

Nº de *Frames*	Falsos Positivos (*FP*)	Falsos Negativos (*FN*)	Verdadeiros Positivos (*VP*)
02	497	15	34
04	228	10	39
06	108	08	41
08	50	04	45
10	26	02	47

A Figura 40 mostra o gráfico da relação entre o número de falsos positivos e o número consecutivo de *frames* processados pelo sistema de DATMO proposto.

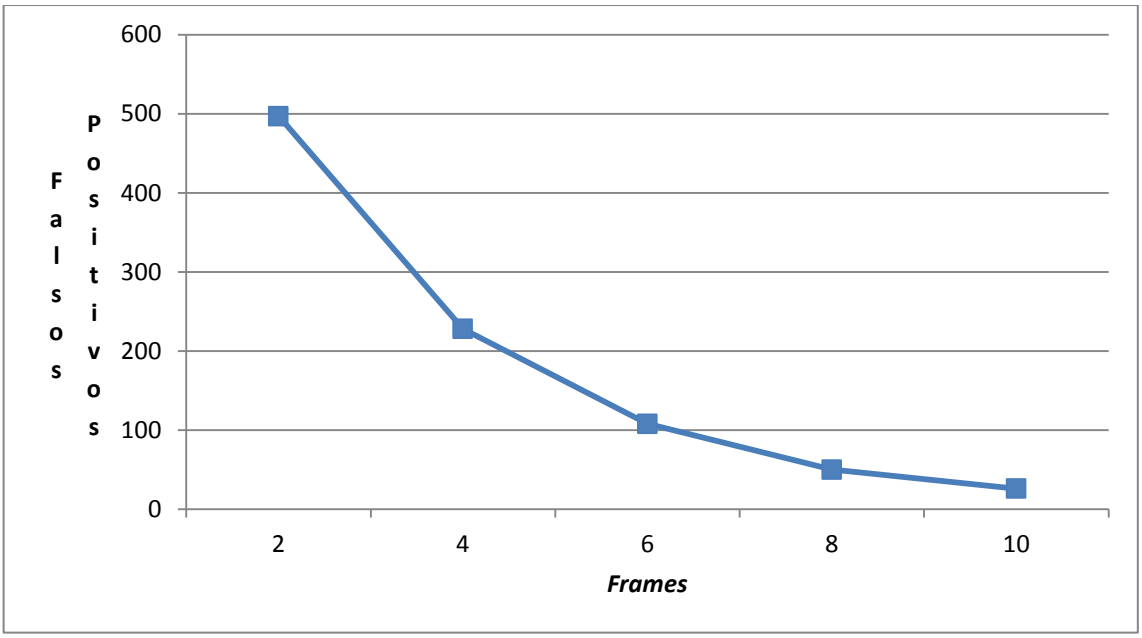


Figura 40: Relação entre o número de falsos positivos e o número de *frames* processados pelo sistema de DATMO proposto

A Figura 41 mostra o gráfico da relação entre o número de falsos negativos e o número consecutivo de *frames* processados pelo sistema de DATMO proposto.

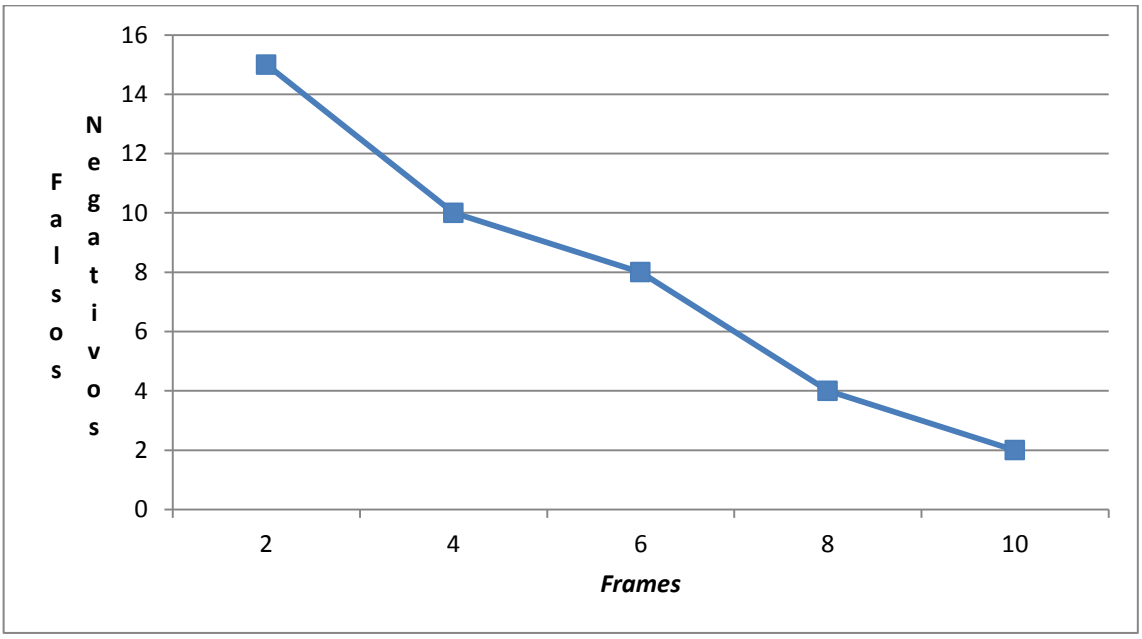


Figura 41: Relação entre o número de falsos negativos e o número de *frames* processados pelo sistema de DATMO proposto

A Figura 42 mostra o gráfico da relação entre o número de verdadeiros positivos e o número consecutivo de *frames* processados pelo sistema de DATMO proposto.

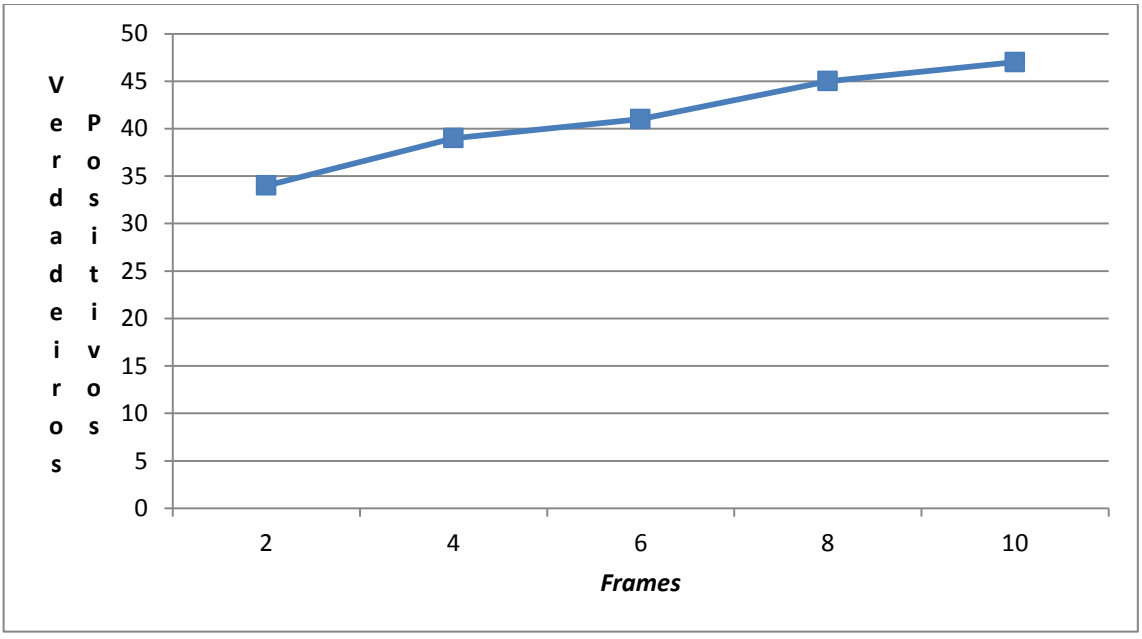


Figura 42: Relação entre o número de verdadeiros positivos e o número de *frames* processados pelo sistema de DATMO proposto

Como observado no gráfico da Figura 40, o número de falsos positivos decresce de acordo com o número de *frames* processados pelo sistema de DATMO. O gráfico da Figura 41 mostra o decréscimo do número de falsos negativos de acordo com o número de *frames* processados pelo sistema de DATMO. O número de verdadeiros positivos cresce de acordo com o número de *frames* processados pelo sistema de DATMO, conforme observado no gráfico da Figura 42. Considerando-se 10 *frames* de processamento pelo sistema de DATMO proposto, obteve-se 26 falsos positivos, 2 falsos negativos e 47 verdadeiros positivos em um domínio de 49 veículos em movimento. O tempo médio para aquisição e processamento de 10 *frames* pelo sistema de DATMO proposto foi de 0,47 segundos.

A Tabela 5 mostra os valores obtidos para os índices de avaliação (precisão e revocação) para um número consecutivo de *frames*.

Tabela 5: Valores obtidos para os índices de avaliação (precisão e revocação) para 2, 4, 6, 8 e 10 *frames*

Nº de *Frames*	**Precisão**	**Revocação**
02	0,06	0,69
04	0,15	0,80
06	0,28	0,84
08	0,47	0,92
10	0,64	0,96

Observando na Tabela 5, é possível verificar um valor de 64% para o índice de precisão após 10 *frames* processados pelo sistema de DATMO proposto. Esse valor é calculado de acordo com a Equação (27), e quanto menor o número de falsos positivos, maior o índice de precisão. Para o índice de revocação obteve-se o valor de 96% após 10 *frames* processados pelo sistema de DATMO proposto (linha 6 da Tabela 5). O valor de revocação é calculado de acordo com a Equação (28), e quanto menor o número de falsos negativos, maior o índice de revocação. Como citado na Seção 1.2, para uma condução autônoma segura, falsos negativos tendem a ser muito perigosos, ao passo que falsos positivos são mais aceitáveis, o que torna um alto índice de revocação um valor desejado.

Como observado na Seção 5.1, o filtro de partículas foi eficiente na discriminação entre objetos estáticos e objetos em movimento ao longo do tempo. Na maioria dos casos, poucas iterações do filtro foram necessárias para corrigir a velocidade estimada para um objeto estático ou em movimento. Observando a Tabela 5, de 2 *frames* para 10 *frames* de processamento, o sistema de DATMO proposto foi capaz de melhorar o índice de precisão em 966,67% (0,64 / 0,06) e o índice de revocação em 39,13% (0,96 / 0,69).

5.3 Discussão

O sistema de DATMO proposto apresentou um bom desempenho na detecção e rastreamento de múltiplos veículos em movimento em um ambiente com um nível razoável de dinamicidade. No entanto, o sistema de DATMO pode ser aperfeiçoado, a fim de detectar e rastrear vários objetos em movimento em ambientes com diversos níveis de dinamicidade.

Uma deficiência do sistema de DATMO proposto é que ele apresenta um fraco desempenho na detecção e rastreamento de pedestres e outros objetos em movimento com baixas velocidades, porque os objetos com velocidade abaixo do limiar de velocidade definido são considerados objetos estáticos. Se o limiar de velocidade definido for reduzido, falsos positivos podem ser detectados e dificilmente falsos negativos podem ser identificados. Apesar de falsos positivos serem mais aceitáveis do que os falsos negativos, o rastreamento de vários falsos positivos pode comprometer o desempenho computacional do sistema. Por outro lado, se o limiar de velocidade definido for aumentado, falsos negativos podem ser identificados, especialmente quando os veículos estão em baixa velocidade, e o número de falsos positivos detectados pode ser reduzido. Uma alternativa para superar esta deficiência seria a implementação de uma etapa de classificação, pela qual os objetos são categorizados em classes de interesse, tais como veículo, pedestre, poste, placa de trânsito, árvore, entre outros. A etapa de classificação otimizaria a etapa de associação de

objetos de diferentes classes, permitiria o uso de modelos de movimento próprios para diversas classes de objeto e permitiria o uso de vários limiares de velocidade.

Outra deficiência do sistema de DATMO proposto é que ele não possui um modelo de transição de estado capaz de estimar o tamanho do objeto, além de sua posição, direção e velocidade, na fase de predição a cada iteração do filtro de partículas. Com mais informações disponibilizadas ao filtro de partículas, o modelo de observação precisaria ser melhorado, a fim de levar em consideração no cálculo dos pesos das partículas várias outras características não observadas no modelo atual. Isto permitiria a geração de caixas delimitadoras de objetos de tamanhos variados e o cômputo de estimativas mais precisas para os estados dos objetos.

Outra deficiência do sistema de DATMO proposto é que, na fase de correção do filtro de partículas, ele calcula os pesos das partículas com base apenas no centro de massa dos objetos. Isto pode acarretar estimativas imprecisas para os estados dos objetos, pois os seus centros de massa podem não ser tão representativos quanto o necessário.

Finalmente, o sistema de DATMO proposto carece de mais experimentos em ambientes com diversos níveis de dinamicidade, a fim de que seja validado de forma mais contundente. Nos experimentos realizados, não foi avaliado o desempenho do sistema de DATMO em ambientes com vários veículos e em diversas direções simultaneamente.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este capítulo apresenta as conclusões e possíveis direções para trabalhos futuros.

6.1 Conclusões

Neste trabalho, foi investigado o problema de detecção e rastreamento de objetos em movimento (*detection and tracking of moving objects* - DATMO) para veículos autônomos e foi proposto um sistema de DATMO para detecção e rastreamento de múltiplos veículos em movimento no ambiente ao redor de um veículo autônomo usando um sensor LIDAR 3D.

O sistema de DATMO proposto opera em três etapas: segmentação, associação e rastreamento. A cada varredura do sensor, após a conversão dos dados do sensor em uma nuvem de pontos 3D com coordenadas globais, na etapa de segmentação, os pontos 3D associados ao plano do solo são removidos; a nuvem de pontos é segmentada em agrupamentos de pontos usando a distância Euclidiana, sendo que cada agrupamento representa um objeto no ambiente; e os agrupamentos relacionados a meios-fios são removidos. Na etapa de associação, os objetos observados na varredura atual do sensor são associados aos mesmos objetos observados em varreduras anteriores usando o algoritmo do vizinho mais próximo (*nearest neighbor*). Finalmente, na etapa de rastreamento, os estados dos objetos são estimados usando um filtro de partículas. Os objetos com velocidade acima de um determinado limiar são considerados veículos em movimento.

O desempenho do sistema de DATMO proposto foi avaliado usando dados de um sensor LIDAR 3D, além de dados de outros sensores, coletados em diferentes datas pelo *Intelligent Autonomous Robotic Automobile* (IARA) ao longo de uma volta pelo anel viário do campus da Universidade Federal do Espírito Santo (UFES). Os resultados experimentais mostraram que o sistema de DATMO proposto foi capaz de detectar e rastrear com bom desempenho múltiplos veículos em movimento no ambien-

te ao redor do veículo autônomo. Entretanto, ele se mostrou inadequado para a detecção e rastreamento de pedestres e outros objetos em movimento com baixas velocidades, porque os objetos com velocidade abaixo de um determinado limiar são considerados objetos estáticos.

6.2 Trabalhos Futuros

O sistema de DATMO proposto abre direções para trabalhos futuros que poderão superar suas deficiências e aperfeiçoar suas capacidades, a fim de detectar e rastrear vários objetos em movimento em ambientes com diversos níveis de dinamicidade.

Uma deficiência do sistema de DATMO proposto é que ele apresenta um fraco desempenho na detecção e rastreamento de pedestres e outros objetos em movimento com baixas velocidades. Para superar esta deficiência, uma direção para trabalhos futuros seria a implementação de uma etapa de classificação dos objetos em classes de interesse. A etapa de classificação permitiria a otimização da etapa de associação [26]. Na etapa de associação, pode-se ignorar uma associação por vizinho mais próximo de objetos de diferentes classes e procurar pelo próximo vizinho mais próximo dentro da esfera especificada. A etapa de classificação permitiria também o uso de modelos de movimento próprios para diferentes classes de objetos, além de diversos limiares de velocidade.

Outra deficiência do sistema de DATMO proposto é que ele não possui um modelo de transição de estado capaz de estimar o tamanho do objeto. Para superar esta deficiência, outra vertente para pesquisas futuras seria a melhoria do modelo de transição de estado, a fim de incorporar o tamanho do objeto em seu estado, e o aperfeiçoamento do modelo de observação, a fim de considerar no cálculo dos pesos das partículas vários outros atributos dos objetos não observados no modelo atual. Isto permitiria a geração de caixas delimitadoras de objetos de tamanhos variados e o cômputo de estimativas mais precisas para os estados dos objetos.

Outra deficiência do sistema de DATMO proposto é que, na fase de correção do filtro de partículas, ele calcula os pesos das partículas com base apenas no centro de massa dos objetos, o que pode acarretar estimativas imprecisas para os estados dos objetos. Outra sugestão para investigações futuras seria a melhoria da abordagem para o cálculo dos pesos das partículas, a fim de permitir o cômputo de estimativas mais precisas para os estados dos objetos. Uma alternativa seria calcular o peso de uma partícula com base na distância entre cada ponto do objeto e a caixa delimitadora do objeto.

Finalmente, outra direção para trabalhos futuros seria a avaliação do sistema de DATMO proposto em ambientes com diversos níveis de dinamicidade.

7 REFERÊNCIAS BIBLIOGRÁFICAS

1. WAISELFIZ, J. J. **Mapa da Violência 2014: Os Jovens do Brasil**. CEBELA-FLACSO. Rio de Janeiro, p. 187. 2014.

2. SIVAK, M.; SCHOETTLE, B. **Mortality from Road Crashes in 193 Countries: A Comparison with Other Leading Causes of Death**. Transportation Research Institute. The University of Michigan. Michigan, p. 43. February 2014. (UMTRI-2014-6).

3. MERCEDES-BENZ Intelligent Drive. On the way to accident-free driving. **Mercedes-Benz**. Disponivel em: <http://www.mercedes-benz-intelligent-drive.com/com/en>. Acesso em: 14 Dezembro 2014.

4. BMW Driver Assitance. **BMW**. Disponivel em: <http://www.bmw.com/com/en/newvehicles/4series/coupe/2013/showroom/driving_assistance_systems/index.html>. Acesso em: 13 Janeiro 2015.

5. AUDI Driver assistance systems and piloted driving. **Audi**. Disponivel em: <https://www.audi-mediaservices.com/publish/ms/content/en/public/hintergrundberichte/2014/01/07/next_generation__/driver_assistance.html>. Acesso em: 13 Janeiro 2015.

6. VOLKSWAGEN Fatigue Detection. **Volkswagen**. Disponivel em: <http://www.volkswagen.com.au/en/technology_and_service/technical-glossary/fatigue-detection.html>. Acesso em: 13 Janeiro 2015.

7. BUEHLER, M.; IAGNEMMA, K.; SINGH, S. (Eds.). **The DARPA Urban Challenge:** Autonomous Vehicles in City Traffic. [S.l.]: Springer, 2009.

8. THE self-driving car logs more miles on new wheels. **Google Official Blog**, 2012. Disponivel em: <http://googleblog.blogspot.hu/2012/08/the-self-driving-car-logs-more-miles-on.html>. Acesso em: 14 Dezembro 2014.

9. JUST press go: designing a self-driving vehicle. **Google Official Blog**, 2014. Disponivel em: <http://googleblog.blogspot.com.br/2014/05/just-press-go-designing-self-driving.html>. Acesso em: 13 Janeiro 2015.

10. AUTONOMOS Labs. **AutoNOMOS Labs**. Disponivel em: <http://www.autonomos.inf.fu-berlin.de/>. Acesso em: 14 Dezembro 2014.

11. AUTONOMOS Labs. **Autonomous Car Navigates the Streets of Berlin**, 2011. Disponivel em: <http://autonomos.inf.fu-berlin.de/news/press-release-92011>. Acesso em: 14 Dezembro 2014.

12. GIZMAG takes a ride in Volvo's most autonomous car yet. **Gizmag**, 2013. Disponivel em: <http://www.gizmag.com/volvo-autonomous-cars/28161/>. Acesso em: 14 Dezembro 2014.

13. NASA and Nissan Join Forces to Build Self-Driving Vehicles for Earth and Space. **Wired**, 2015. Disponivel em: <http://www.wired.com/2015/01/nasa-nissan-join-forces-build-self-driving-vehicles-earth-space/>. Acesso em: 22 Janeiro 2015.

14. THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic Robotics**. Cambridge, Massachusetts: MIT Press, 2006.

15. PETROVSKAYA, A. et al. Awareness of Road Scene Participants for Autonomous Driving. In: _____ **Handbook of Intelligent Vehicles**. London: Springer-Verlag London Ltd, 2012. p. 1383-1432.

16. RADAELLI, R. R. et al. **A Motion Planner for Car-Like Robots Based on Rapidly-Exploring Random Trees**. 14th Ibero-American Conference on Artificial Intelligence. Santiago, Chile: Springer International Publishing. 2014. p. 469-480.

17. PETROVSKAYA, A.; THRUN, S. Model based vehicle detection and tracking for autonomous urban driving. **Autonomous Robots - Springer US**, 26, 01 April

2009. 123-139.

18. BECKER, M. et al. 2D laser-based probabilistic motion tracking in urban-like environments. **Journal of the Brazilian Society of Mechanical Sciences and Engineering**, Rio de Janeiro, v. 31, n. 2, p. 83-94, April-June 2009.

19. WANG, C.-C. et al. Simultaneous Localization, Mapping and Moving Object Tracking. **The International Journal of Robotics Research**, v. 26, n. 09, p. 889–916, September 2007.

20. VU, T.-D.; AYCARD, O.; APPENRODT, N. **Online Localization and Mapping with Moving Object Tracking in Dynamic Outdoor Environments**. IEEE Intelligent Vehicles Symposium. Istanbul, Turkey: IEEE. 2007. p. 190-195.

21. WANG, C.-C.; THORPE, C.; THRUN, S. **Online simultaneous localization and mapping with detection and tracking of moving objects:** theory and results from a ground vehicle in crowded urban areas. IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA '03. Taipei, Taiwan: IEEE. 14-19 Sept. 2003. p. 842 - 849.

22. MACLACHLAN, R.; MERTZ, C. **Tracking of Moving Objects from a Moving Vehicle Using a Scanning Laser Rangefinder**. Intelligent Transportation Systems. [S.l.]: IEEE. September, 2006. p. 301-306.

23. VU, T.-D.; AYCARD, O. **Laser-based detection and tracking moving objects using data-driven Markov chain Monte Carlo**. IEEE International Conference on Robotics and Automation, 2009. ICRA '09. Kobe, Japan: IEEE. DOI:10.1109/ROBOT.2009.5152805. 2009. p. 3800 - 3806.

24. KAESTNER, R. et al. **Generative object detection and tracking in 3D range data**. IEEE International Conference on Robotics and Automation (ICRA). Saint Paul, MN: IEEE. DOI: 10.1109/ICRA.2012.6224585. 14-18 May 2012. p. 3075 - 3081.

25. AZIM, A.; AYCARD, O. **Layer-based supervised classification of moving objects in outdoor dynamic environment using 3D laser scanner**. IEEE Intelligent Vehicles Symposium Proceedings. Dearborn, MI: IEEE. DOI: 10.1109/IVS.2014.6856558. 8-11 June 2014. p. 1408 - 1414.

26. AZIM, A.; AYCARD, O. Detection, classification and tracking of moving objects in a 3D environment. **Intelligent Vehicles Symposium (IV), 2012 IEEE**, Alcala de Henares, p. 802 - 807, 3-7 June 2012. ISSN DOI: 10.1109/IVS.2012.6232303.

27. THRUN, S. et al. Stanley: The robot that won the DARPA Grand Challenge. **Journal of Field Robotics**, v. 23, n. 9, p. 661–692, September 2006. ISSN DOI: 10.1002/rob.20147.

28. BAKER, C. R.; DOLAN, J. M. Street smarts for boss. **IEEE Robotics & Automation Magazine**, v. 16, n. 1, p. 78 - 87, March 2009. ISSN DOI: 10.1109/MRA.2008.931629.

29. TEAM, S. R. Stanford's Robotic Vehicle "Junior:" Interim Report. **DARPA Urban Challenge**, 12 April 2007. Disponivel em: <http://archive.darpa.mil/grandchallenge/TechPapers/Stanford.pdf>. Acesso em: 14 Dezembro 2014.

30. LIGGINS, M. I. et al. Distributed fusion architectures and algorithms for target tracking. **Proceedings of the IEEE**, v. 85, n. 1, p. 95 - 107, Jan. 1997. ISSN DOI: 10.1109/JPROC.1997.554211.

31. NASHASHIBI, F.; BARGETON, A. **Laser-based vehicles tracking and classification using occlusion reasoning and confidence estimation**. IEEE Intelligent Vehicles Symposium. Eindhoven: IEEE. DOI: 10.1109/IVS.2008.4621244. 4-6 June 2008. p. 847 - 852.

32. BAIG, Q. et al. **Fusion between laser and stereo vision data for moving objects tracking in intersection like scenario**. IEEE Intelligent Vehicles

Symposium (IV). Baden-Baden: IEEE. DOI: 10.1109/IVS.2011.5940576. 5-9 June 2011. p. 362 - 367.

33. SCHULZ, D. et al. **Tracking multiple moving objects with a mobile robot**. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. [S.l.]: IEEE. DOI: 10.1109/CVPR.2001.990499. 2001. p. I-371 - I-377 vol.1.

34. ALMEIDA, A.; ALMEIDA, J.; ARAUJO, R. **Real-Time Tracking of Moving Objects Using Particle Filters**. Proceedings of the IEEE International Symposium on Industrial Electronics. Dubrovnik, Croatia: IEEE. DOI: 10.1109/ISIE.2005.1529124. June 20-23, 2005. p. 1327 - 1332.

35. SATO, S. et al. **Multilayer lidar-based pedestrian tracking in urban environments**. IEEE Intelligent Vehicles Symposium (IV). San Diego, CA: IEEE. DOI: 10.1109/IVS.2010.5548135. 21-24 June 2010. p. 849 - 854.

36. WU, M.; SUN, J.-Y. **Moving Object Detecting and Tracking with Mobile Robot Based on Extended Kalman Filter in Unknown Environment**. International Conference on Machine Vision and Human-Machine Interface (MVHI). Kaifeng, China: IEEE. DOI: 10.1109/MVHI.2010.88. 24-25 April 2010. p. 64 - 67.

37. RICHTER, E.; SCHUBERT, R.; WANIELIK, G. **Radar and vision based data fusion - Advanced filtering techniques for a multi object vehicle tracking system**. IEEE Intelligent Vehicles Symposium. Eindhoven: IEEE. DOI: 10.1109/IVS.2008.4621245. 4-6 June 2008. p. 120 - 125.

38. VANPOPERINGHE, E.; WAHL, M.; NOYER, J.-C. **Model-based detection and tracking of vehicle using a scanning laser rangefinder:** A particle filtering approach. IEEE Intelligent Vehicles Symposium (IV). Alcala de Henares, Spain: IEEE. DOI: 10.1109/IVS.2012.6232265. 3-7 June 2012. p. 1144 - 1149.

39. COUÉ, C. et al. Bayesian Occupancy Filtering for Multitarget Tracking: an Automotive Application. **The International Journal of Robotics Research, SAGE Publications**, v. 25, n. 1, p. 19-30, January 2006. ISSN DOI: 10.1177/0278364906061158.

40. DOUILLARD, B. et al. **On the segmentation of 3d lidar point-clouds**. IEEE International Conference on Robotics and Automation (ICRA). Shanghai: IEEE. DOI: 10.1109/ICRA.2011.5979818. 9-13 May 2011. p. 2798 - 2805.

41. FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. **Magazine Communications of the ACM**, New York, NY, USA, v. 24, n. 6, p. 381-395, June 1981. ISSN DOI: 10.1145/358669.358692.

42. HUANG, W.; GONG, X. **Fusion Based Holistic Road Scene Understanding**. Cornell University Library. arXiv.org. [S.l.], p. 14. 29 Jun 2014. (arXiv:1406.7525 [cs.CV]).

43. HENRY, P. et al. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. **The International Journal of Robotics Research**, v. 31, n. 5, p. 647-663, April 2012. ISSN DOI: 10.1177/0278364911434148.

44. BORCS, A.; NAGY, B.; BENEDEK, C. **On board 3D object perception in dynamic urban scenes**. IEEE 4th International Conference on Cognitive Infocommunications (CogInfoCom). Budapest: IEEE. DOI: 10.1109/CogInfoCom.2013.6719301. 2-5 Dec. 2013. p. 515 - 520.

45. HIMMELSBACH, M. et al. **LIDAR-based 3D Object Perception**. In Proceedings of 1st International Workshop on Cognition for Technical Systems. Munich: [s.n.]. 2008. p. 7.

46. SPINELLO, L. et al. **A Layered Approach to People Detection in 3D Range**

Data. Twenty-Fourth AAAI Conference on Artificial Intelligence. [S.l.]: AAAI Publications. 2010. p. 6.

47. MACLACHLAN, R. **Tracking Moving Objects From a Moving Vehicle Using a Laser Scanner**. The Robotics Institute. Carnegie Mellon University. Pittsburgh, PA., p. 37. 23 June 2005. (CMU-RI-TR-05-07).

48. GOLOVINSKIY, A.; KIM, V. G.; FUNKHOUSER, T. **Shape-based recognition of 3d point clouds in urban environments**. International Conference on Computer Vision (ICCV). [S.l.]: [s.n.]. September 2009. p. 8.

49. ESTER, M. et al. **A density-based algorithm for discovering clusters in large spatial databases with noise**. in Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96). [S.l.]: AAAI Press. 1996. p. 226--231.

50. LI, W. et al. An approach of laser-based vehicle monitor. **Applied Mathematics and Computation**, v. 185, n. 2, p. 953–962, 15 February 2007. ISSN doi:10.1016/j.amc.2006.07.032.

51. L-DBSCAN: A Fast Hybrid Density Based Clustering Method. IEEE Proceedings of the 18th International Conference on Pattern Recognition. [S.l.]: IEEE Computer Society Washington, DC, USA. DOI: 10.1109/ICPR.2006.741. 2006. p. 912-915.

52. POINT Cloud Library (PCL). **Euclidean Cluster Extraction**. Disponivel em: <http://pointclouds.org/documentation/tutorials/cluster_extraction.php>. Acesso em: 14 Dezembro 2014.

53. RUSU, R. B.; COUSINS, S. **3D is here:** Point Cloud Library (PCL). IEEE International Conference on Robotics and Automation (ICRA). Shanghai, China: IEEE. DOI: 10.1109/ICRA.2011.5980567. 9-13 May 2011. p. 1 - 4.

54. BOUCHER, S. **Obstacle Detection and Avoidance Using TurtleBot Platform and XBox Kinect**. Department of Computer Science. Rochester Institute of Technology. [S.l.], p. 56. August 9, 2012.

55. HAUSMAN, K. et al. **Tracking-based interactive segmentation of textureless objects**. IEEE International Conference on Robotics and Automation (ICRA). Karlsruhe: IEEE. DOI: 10.1109/ICRA.2013.6630713. 6-10 May 2013. p. 1122 - 1129.

56. TELLAECHE, A.; MAURTUA, I. **6DOF pose estimation of objects for robotic manipulat ion. A review of different options**. IEEE Emerging Technology and Factory Automation (ETFA). Barcelona, Spain: IEEE. DOI: 10.1109/ETFA.2014.7005077. 16-19 Sept. 2014. p. 1 - 8.

57. WANG, D. Z.; POSNER, I.; NEWMAN, P. **What could move? Finding cars, pedestrians and bicyclists in 3D laser data**. IEEE International Conference on Robotics and Automation (ICRA). Saint Paul, MN: IEEE. DOI: 10.1109/ICRA.2012.6224734. 14-18 May 2012. p. 4038 - 4044.

58. ALI, H.; FIGUEROA, N. **Segmentation and Pose Estimation of Planar Metallic Objects**. Ninth Conference on Computer and Robot Vision (CRV). Toronto, ON: IEEE. DOI: 10.1109/CRV.2012.56. 28-30 May 2012. p. 376 - 382.

59. LITOMISKY, K.; BHANU, B. Removing Moving Objects from Point Cloud Scenes. In: _____ **Advances in Depth Image Analysis and Applications**. DOI: 10.1007/978-3-642-40303-3_6. ed. Tsukuba, Japan: Springer Berlin Heidelberg, v. 7854, 2013. p. 50-58.

60. COX, I. J. A Review of Statistical Data Association Techniques for Motion Correspondence. **International Journal of Computer Vision**, v. 10, n. 1, p. 53-66 , fev. 1993. ISSN DOI: 10.1007/BF01440847.

61. SCHULZ, D. et al. People Tracking with a Mobile Robot Using Sample-Based

Joint Probabilistic Data Association Filters. **International Journal of Robotics Research (IJRR)**, v. 22, n. 2, p. 34, February 2003.

62. JAWARD, M. H. et al. **A data association algorithm for multiple object tracking in video sequences**. The IEE Seminar on Target Tracking: Algorithms and Applications, (Ref. No. 2006/11359). Birmingham: IET. 7-8 March 2006. p. 129 - 136.

63. ELFRING, J.; JANSSE, R.; MOLENGRAFT, R. V. D. **Data Association and Tracking: A Literature Survey**. RoboEarth Project. Eindhoven, The Netherlands, p. 42. April 1, 2010. (FP7-ICT-248942).

64. DIAS, S. S. **Avaliação do método MHT em cenários com múltiplos alvos**. Dissertação de Mestrado em Engenharia Elétrica - Instituto Tecnológico de Aeronáutica. São José dos Campos, p. 136. 2008.

65. VIDAL, F. D. B. **Rastreamento Visual de Objetos Utilizando Métodos de Similaridade de Regiões e Filtragem Estocástica**. Tese de Doutorado em Engenharia Elétrica, Universidade de Brasília. Brasília, p. 99. 2009. (PPGENE.TD-041/09).

66. BLACKMAN, S. S. Multiple hypothesis tracking for multiple target tracking. **IEEE Aerospace and Electronic Systems Magazine**, El Segundo, CA, USA, v. 19, n. 1, p. 5 - 18, Jan 2004. ISSN IEEE. DOI: 10.1109/MAES.2004.1263228.

67. LIM, Y.-C. et al. Stereo-Based Tracking-by-Multiple Hypotheses Framework for Multiple Vehicle Detection and Tracking. **International Journal of Advanced Robotic Systems**, v. 10, p. 13, May 2013. ISSN DOI: 10.5772/56688.

68. MEJIA-INIGO, R. et al. **Vehicle tracking based on multiple hypotheses**. 8th International Conference on Electrical Engineering Computing Science and Automatic Control (CCE). Merida City: IEEE. DOI: 10.1109/ICEEE.2011.6106598. 26-28 Oct. 2011. p. 1 - 6.

69. CHO, K.; BAEG, S.; PARK, S. Object tracking with enhanced data association using a 3D range sensor for an unmanned ground vehicle. **Journal of Mechanical Science and Technology**, v. 28, n. 11, p. 4381-4388, November 2014. ISSN DOI: 10.1007/s12206-014-1005-6.

70. KAKINUMA, K. et al. **Cooperative pedestrian tracking by multi-vehicles in GPS-denied environments**. IEEE Proceedings of SICE Annual Conference (SICE). Akita: IEEE. ISBN: 978-1-4673-2259-1. 20-23 Aug. 2012. p. 211 - 214.

71. KALMAN, R. E. A new approach to linear filtering and prediction problems. **Transactions of the ASME - Journal of Basic Engineering**, v. 82, n. Series D, p. 35-45, 1960.

72. AIUBE, F. A. L. **Modelagem dos preços futuros de commodities: abordagem pelo filtro de partículas**. Tese (doutorado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Industrial. Rio de Janeiro, p. 183. 2005.

73. DOUCET, A.; GODSILL, S.; ANDRIEU, C. On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. **STATISTICS AND COMPUTING**, v. 10, n. 3, p. 197-208, 2000.

74. GORDON, N. J.; SALMOND, D. J.; SMITH, A. F. M. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. **IEEE Proceedings F Radar and Signal Processing**, v. 140, n. 2, p. 107 - 113, Apr 1993. ISSN ISSN: 0956-375X.

75. ARULAMPALAM, M. S. et al. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. **IEEE Transactions on Signal Processing**, v. 50, n. 2, p. 174 - 188, Feb 2002. ISSN DOI: 10.1109/78.978374.

76. THRUN, S. **Particle Filters in Robotics**. in Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI). [S.l.], p. 9. 2002.

77. CAPPE, O.; GODSILL, S. J.; MOULINES, E. An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo. **Proceedings of the IEEE**, v. 95, n. 5, p. 899 - 924, May 2007. ISSN DOI: 10.1109/JPROC.2007.893250.

78. BRANDÃO, B. C. **Rastreando a Mão com o Filtro de Partículas com Hierarquia de Subespaços**. Dissertação de Mestrado. Universidade Estadual de Campinas, Instituto de Computação. Campinas, SP, p. 88. 2006.

79. SMITH, A. F. M.; GELFANDB, A. E. Bayesian Statistics without Tears: A Sampling–Resampling Perspective. **The American Statistician**, v. 46, n. 2, p. 84-88, 1992. ISSN DOI: 10.1080/00031305.1992.10475856.

80. WILLIAMS, D. N. Dealing with Data Overload in the Scientific Realm. **Science and Technology Review. Lawrence Livermore National Laboratory**, Livermore, CA, USA, p. 4-11, January/February 2013.

81. BOUTTEFROY, P. L. M. et al. **Vehicle Tracking Using Projective Particle Filter**. Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance. Genova : IEEE. DOI: 10.1109/AVSS.2009.60. 2-4 Sept. 2009. p. 7 - 12.

82. STEUX, B.; ABRAMSON, Y. **Robust real-time on-board vehicle tracking system using particles filter**. IFAC Symposium on Intelligent Autonomous Vehicles. [S.l.]: [s.n.]. July 2004. p. 6.

83. BREITENSTEIN, M. D. et al. **Robust tracking-by-detection using a detector confidence particle filter**. IEEE 12th International Conference on Computer Vision. Kyoto: IEEE. DOI: 10.1109/ICCV.2009.5459278. Sept. 29-Oct. 2 2009. p. 1515 - 1522.

84. AL-MUTIB, K. et al. **Motion periodicity based pedestrian detection and particle filter based pedestrian tracking using stereo vision camera**. IEEE 19th International Conference Mechatronics and Machine Vision in Practice

(M2VIP). Auckland: IEEE. 28-30 Nov. 2012. p. 32-37.

85. DANESCU, R.; NEDEVSCHI, S. A Particle-Based Solution for Modeling and Tracking Dynamic Digital Elevation Maps. **IEEE Transactions on Intelligent Transportation Systems**, v. 15, n. 3, p. 1002 - 1015, June 2014. ISSN IEEE. DOI: 10.1109/TITS.2013.2291447.

86. MUTZ, F. W. **Um Sistema para Mapeamento de Grandes Regiões usando GraphSLAM**. Dissertação de Mestrado. Programa de Pós-Graduação em Informática. UFES. Vitória, p. 112. 2013.

87. LIU, Z.; WANG, J.; LIU, D. A New Curb Detection Method for Unmanned Ground Vehicles Using 2D Sequential Laser Data. **Sensors 2013**, v. 13, n. 1, p. 1102-1120, 16 January 2013. ISSN DOI:10.3390/s130101102.

88. **TORC Robotics**, 2015. Disponivel em: <http://www.torcrobotics.com>. Acesso em: 13 Janeiro 2015.

89. VELODYNE Lidar. **Velodyne Acoustics, Inc.**, 2015. Disponivel em: <http://velodynelidar.com/lidar/lidar.aspx>. Acesso em: 12 Janeiro 2015.

90. MONTEMERLO, M.; ROY, N.; THRUN, S. **Perspectives on standardization in mobile robot programming:** the Carnegie Mellon Navigation (CARMEN) Toolkit. IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2003. [S.l.]: IEEE. 27-31 Oct. 2003. p. 2436 - 2441 vol.3.

91. SIMMONS, R. The inter-process communication (IPC) system. Disponivel em: <http://www.cs.cmu.edu/afs/cs/project/TCA/www/ipc/ipc.html>. Acesso em: 12 Fevereiro 2015.