UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

THIAGO HENRIQUE DE OLIVEIRA

Segmentação dinâmica de objetos aplicada à odometria visual

Goiânia
2024

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO (TECA) PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES

E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico

[X] Dissertação [ ] Tese [ ] Outro*:_____

*No caso de mestrado/doutorado profissional, indique o formato do Trabalho de Conclusão de Curso, permitido no documento de área, correspondente ao programa de pós-graduação, orientado pela legislação vigente da CAPES.

Exemplos: Estudo de caso ou Revisão sistemática ou outros formatos.

2. Nome completo do autor

Thiago Henrique de Oliveira

3. Título do trabalho

Segmentação dinâmica de objetos aplicada à odometria visual

4. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador)

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante:
a) consulta ao(à) autor(a) e ao(à) orientador(a);
b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação.
O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Gustavo Teodoro Laureano**, **Professor do Magistério Superior**, em 06/12/2024, às 09:49, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

Documento assinado eletronicamente por **Thiago Henrique De Oliveira**, **Discente**, em 09/12/2024, às 16:10, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4958652** e o código CRC **3E9E28B1**.

Referência: Processo nº 23070.046159/2024-28 SEI nº 4958652

THIAGO HENRIQUE DE OLIVEIRA

Segmentação dinâmica de objetos aplicada à odometria visual

Dissertação apresentada ao Programa de Pós–Graduação em Ciência da Computação, do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Ciência da Computação
Linha de pesquisa: Sistemas Inteligentes e Aplicações.

Orientador: Prof. Gustavo Teodoro Laureano

Goiânia
2024

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Oliveira, Thiago Henrique de
Segmentação dinâmica de objetos aplicada à odometria visual [manuscrito] / Thiago Henrique de Oliveira. - 2024.
LXVII, 66 f.: il.

Orientador: Prof. Dr. Gustavo Teodoro Laureano .
Dissertação (Mestrado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação, Goiânia, 2024.
Bibliografia.

1. Odometria Visual. 2. Segmentação. 3. Ambientes dinâmicos. I. , Gustavo Teodoro Laureano, orient. II. Título.

CDU 004

UNIVERSIDADE FEDERAL DE GOIÁS

INSTITUTO DE INFORMÁTICA

ATA DE DEFESA DE DISSERTAÇÃO

Ata nº **27** da sessão de Defesa de Dissertação de **Thiago Henrique de Oliveira**, que confere o título de Mestre em Ciência da Computação, na área de concentração em Ciência da Computação.

Aos dois dias do mês de outubro de dois mil e vinte e quatro, a partir das nove horas, via webconferência, realizou-se a sessão pública de Defesa de Dissertação intitulada **"Segmentação dinâmica de objetos aplicada à odometria visual"**. Os trabalhos foram instalados pelo Orientador, Professor Doutor Gustavo Teodoro Laureano (INF/UFG) com a participação dos demais membros da Banca Examinadora: Professor Doutor Fernando Santos Osório (ICMC/USP), membro titular externo; Professor Doutor Anderson da Silva Soares (INF/UFG), membro titular interno. A realização da banca ocorreu por meio de videoconferência. Durante a arguição os membros da banca não fizeram sugestão de alteração do título do trabalho. A Banca Examinadora reuniu-se em sessão secreta a fim de concluir o julgamento da Dissertação, tendo sido o candidato **aprovado** pelos seus membros. Proclamados os resultados pelo Professor Doutor Gustavo Teodoro Laureano, Presidente da Banca Examinadora, foram encerrados os trabalhos e, para constar, lavrou-se a presente ata que é assinada pelos Membros da Banca Examinadora, aos dois dias do mês de outubro de dois mil e vinte e quatro.

TÍTULO SUGERIDO PELA BANCA

Documento assinado eletronicamente por **Anderson Da Silva Soares**, **Professor do Magistério Superior**, em 02/10/2024, às 11:22, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

Documento assinado eletronicamente por **Gustavo Teodoro Laureano**, **Professor do Magistério Superior**, em 02/10/2024, às 11:30, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

Documento assinado eletronicamente por **Fernando Santos Osório**, **Usuário Externo**, em 02/10/2024, às 11:56, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Thiago Henrique De Oliveira**, **Discente**, em 02/10/2024, às 17:10, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4846016** e o código CRC **D57ADEF9**.

Resumo

Oliveira, Thiago Henrique de. **Segmentação dinâmica de objetos aplicada à odometria visual**. Goiânia, 2024. 66p. Dissertação de Mestrado. Programa de Pós-Graduação em Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás.

A presença de objetos dinâmicos em uma cena pode prejudicar significativamente o desempenho de métodos de odometria visual. Mesmo com o uso de métodos robustos, nem sempre é possível evitar outliers e interferências na estimação do movimento da câmera. Esse tipo de objeto introduz pontos característicos cujo movimento não condiz com o movimento real realizado pela câmera. Para filtrar esses objetos, este trabalho apresenta uma arquitetura de rede neural que combina imagens em RGB e fluxo óptico para segmentar regiões que apresentam objetos em movimento mesmo com o movimento da própria câmera. Para viabilizar o treinamento da rede, é apresentada uma metodologia rápida de anotação de *datasets* de detecção de objetos para adicionar máscaras semânticas de objetos em movimento a 98491 imagens de um conjunto de dados de navegação urbana. A rede neural proposta foi treinada e avaliada com esses dados, e se mostrou adequada para uso como filtro de objetos dinâmicos em tarefas de odometria visual. Para avaliar o modelo proposto, são apresentadas comparações de algoritmos de odometria visual com e sem o uso da filtragem. A partir dos resultados obtidos por este trabalho, a identificação e filtragem de objetos dinâmicos em uma imagem se apresenta como uma etapa fundamental da tarefa de odometria visual, sendo essencial para aplicações com a presença de objetos dinâmicos.

Palavras–chave

Odometria Visual, Segmentação, Ambientes dinâmicos

Abstract

Oliveira, Thiago Henrique de. **Dynamic object segmentation applied to visual odometry**. Goiânia, 2024. 66p. MSc. Dissertation. Programa de Pós-Graduação em Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás.

The presence of dynamic objects in a scene can significantly impair the performance of visual odometry methods. Even with the use of robust methods, it is not always possible to avoid outliers and interferences in the estimation of the camera's movement. This type of object introduces characteristic points whose movement does not align with the actual movement performed by the camera. To filter these objects, this work presents a neural network architecture that combines RGB images and optical flow to segment regions that exhibit moving objects, even while the camera itself moves. To enable the training of the network, a methodology for quick annotation of object detection datasets is presented to add semantic masks of moving objects to 98,491 images of an urban navigation dataset. The proposed neural network was trained and evaluated with these data and proved adequate for use as a dynamic object filter in visual odometry tasks. To evaluate the proposed model, comparisons of visual odometry algorithms with and without the use of filtering are presented. Based on the results obtained in this work, the identification and filtering of dynamic objects in an image emerges as a fundamental step in the task of visual odometry, being essential for applications involving the presence of dynamic objects.

Keywords

Visual odometry, Segmentation, Dynamic Environments

Sumário

CAPÍTULO 1

Introdução

A odometria visual, ou *Visual Odometry* (VO) em inglês, é o processo de estimar o movimento de um agente, como um veículo ou pessoa, usando imagens capturadas por uma ou mais câmeras. A odometria visual consiste em estimar de forma incremental a posição e orientação do veículo ao examinar as mudanças que o movimento causa nas imagens capturadas por sua câmera. Ela pode ser aplicada em robótica, realidade aumentada, e automotiva. [Scaramuzza e Fraundorfer 2011].

A precisão da odometria visual pode ser afetada por condições adversas, como variações de iluminação, reflexos, falta de textura no ambiente e objetos em movimento. A VO geralmente presume que o ambiente é estático, ou seja, todo movimento observado pela câmera é causado pelo seu próprio movimento. Essa suposição leva a resultados imprecisos quando há outros objetos em movimento no ambiente, pois o sistema não consegue distinguir entre o movimento da câmera e o movimento de outros objetos móveis no ambiente. Métodos de odometria convencionais utilizam de métodos de remoção de *outliers* como o RANSAC [Fischler e Bolles 1981] para mitigar essa limitação, mas eles podem não ser capazes de descartar esses pontos caso a proporção de objetos em movimento no ambiente seja muito grande.

Este problema é particularmente importante para aplicações em veículos autônomos para ambientes de tráfego urbano e ambientes fechado onde há uma grande quantidade de veículos, pessoas e outros objetos se movimentado. Aprimorar a precisão da odometria visual nessas condições torna mais factível a aplicação de veículos autônomos e robôs em ambientes mais complexos.

O objetivo deste trabalho é desenvolver um método para a filtragem de pontos dinâmicos utilizando imagens provenientes de um sistema monocular, uma única câmera, para fornecer dados mais consistentes para algoritmos de odometria visual. Esse filtro é integrado em um sistema de odometria visual e testado em ambientes com grande quantidade de veículos em movimento, para avaliar seu impacto na acurácia da estimativa.

1.1 Proposta e objetivo do trabalho

A revisão da literatura indicou que o uso de imagem RGB combinado com o fluxo óptico em redes neurais de segmentação de objetos em movimento pode ser eficaz para obter uma acurácia melhor em odometria visual em ambiente dinâmicos.

Nesse contexto, este trabalho propõe a integração de imagens RGB e fluxo óptico para o treinamento de modelos neurais de segmentação de regiões dinâmicas na cena em favor da seleção de regiões mais favoráveis para o cálculo de odometria visual. Vale notar que alcançar desempenho em tempo real não é um dos objetivos deste trabalho, e todos os experimentos com odometria foram realizados de forma offline.

Um desafio encontrado é a baixa quantidade de *datasets* adequados para ambiente urbano com anotações de segmentação que diferenciem entre objetos estáticos e dinâmicos. O *dataset* KITTI-Motion [Vertens, Valada e Burgard 2017], por exemplo, disponibiliza dados de segmentação e movimento, mas é composto de poucas imagens. O WHUVID [Chen et al. 2022] é um *dataset* de navegação urbana com grande quantidade de imagens, em ambientes variados com grande quantidade de carros em movimento, mas sua anotação não dispõe de informações de segmentação e classificação entre objetos estáticos e dinâmico, as quais são necessárias para o treinamento da rede proposta neste trabalho. Por esse motivo, um objetivo específico a ser alcançado é a a construção de uma metodologia de anotação para complementar o *dataset* WHUVID, tornando-o adequado para o treinamento da rede neural.

Para alcançar esse objetivo, foi desenvolvido um *script* de anotação rápido que utiliza de métodos de textittracking de objetos para permitir que as anotações sejam feitas de objeto a objeto, e não de imagem por imagem. O Segment Anything [Kirillov et al. 2023] foi utilizado para gerar máscaras de segmentação a partir das *bounding boxes* presentes no *dataset*. Essas ferramentas foram utilizadas para anotar quase 100 mil imagens do WHUVID.

Para alcançar o objetivo proposto, foi definido a seguinte lista de objetivos específicos:

1. Seleção de *dataset* para treinamento de segmentação;
2. Desenvolvimento de sistema de anotação rápida de *labels* de movimento;
3. Anotação do *dataset* WHUVID;
4. Desenvolvimento da arquiteturas de segmentação;
5. Treinamento da rede para a segmentação de objetos dinâmicos, combinando imagens RGB e Fluxo Óptico;
6. Avaliação dos resultados da segmentação;
7. Integração do filtro em um sistema de odometria visual monocular;
8. Avaliação da odometria visual com e sem o filtro proposto.

As principais contribuições deste trabalho são:

1. Um método de anotação rápida para acrescentar *labels* por objeto que distinguem entre objetos dinâmicos e estáticos.
2. Anotação de quase 100 mil imagens com mascara de segmentação de objetos estáticos e dinâmicos para o *dataset* WHUVID
3. Uma rede neural treinada para segmentar objetos dinâmicos em cenários urbanos
4. Avaliações do impacto da filtragem de pontos dinâmicos na estimativa da odometria visual

1.2 Organização deste trabalho

O Capítulo 2 apresenta alguns trabalhos relacionados com esta pesquisa e discute algumas soluções que inspiraram a proposta deste trabalho. O Capítulo 3 descreve os fundamentos teóricos importantes para a proposta. É apresentada a Odometria Visual e seu *pipeline* básico, a definição de segmentação semântica e algumas das redes neurais de segmentação mais usadas, uma noção de Fluxo Óptico e exemplos de algoritmos, e também uma introdução sobre rastreamento de múltiplos objetos em vídeo e alguns algoritmos de *tracking*. O Capítulo 4 apresenta a metodologia proposta por esse trabalho, iniciando pela agregação de informações aos *datasets* escolhidos e a arquitetura de rede neural para a segmentação de objetos dinâmicos. O Capítulo 5 descreve os experimentos realizados para avaliar a segmentação desenvolvida. E, por fim, o Capítulo 6 apresenta as conclusões do trabalho.

CAPÍTULO 2

Trabalhos Relacionados

A odometria visual é uma técnica bem estabelecida na área de visão robótica, pois ela permite estimar o movimento realizado por uma câmera analisando imagens consecutivas de uma cena, o que consiste em uma tarefa fundamental dessa área. Diversos algoritmos de odometria visual são encontrados na literatura, variando entre abordagens baseadas em correspondência de características ou em aparência [Scaramuzza e Fraundorfer 2011]. O MonoSLAM [Davison et al. 2007], por exemplo, segue um fluxo de processamento amplamente difundido na área, construindo um mapa 3D esparso usando uma abordagem probabilística. Libvisio2 [Geiger, Ziegler e Stiller 2011] disponibiliza a implementação de odometria visual para sistemas monoculares e estéreo, usando a estratégia de reduzir o erro de reprojeção de pontos correspondentes. O sistema ORB-SLAM [Mur-Artal Raúl; Montiel e Tardós 2015] utiliza características ORB para rastreamento, mapeamento, relocalização e *loop-closure*, oferecendo uma técnica eficiente e robusta para odometria visual monocular. O trabalho subsequente, o ORB-SLAM2 [Mur-Artal e Tardós 2017] estende o ORB-SLAM para sistemas monocular, estéreo e RGB-D, resultando em um método de SLAM versátil e amplamente difundido em aplicações mais atuais.

O uso de métodos robustos de estimação é importante em situações em que a correspondência de características falha. Para lidar com a presença de *outliers*, muitos trabalhos optam pelo uso do algoritmo RANSAC [Fischler e Bolles 1981], como por exemplo o ORB-SLAM e o Libvisio2. Com esse algoritmo é possível selecionar o grupo de pontos que mantém uma coerência de movimento, independente da presença de pontos ruidosos. No entanto, o ruído não se dá apenas pela correspondência errada de pontos. De modo geral, qualquer informação que difere da coerência estabelecida pelo movimento da própria câmera pode ser classificada como *outilers*, como é o caso em cenários que admitem objetos dinâmicos.

Em ambientes dinâmicos, pontos que pertencem a um mesmo objeto compartilham o mesmo tipo de movimento entre si, mas divergem do movimento da própria câmera. Neste contexto, o algoritmo RANSAC pode descartar esses pontos caso não se-

jam a maioria, entretanto, em situações em que a quantidade de *outliers* é alta, outros métodos robustos devem ser utilizados.

Algumas soluções utilizam sensores inerciais (IMUs) para complementar e melhorar a odometria visual. Um exemplo é o Vins-Mono [Qin, Li e Shen 2018], que implementa um procedimento de inicialização robusto e um método de otimização não linear para fundir medições de IMU e observações de pontos característicos, promovendo alta precisão. Outro exemplo é o MSCKF [Mourikis e Roumeliotis 2007], que deriva um modelo de medidas capaz de expressar as restrições geométricas que surgem quando uma característica estática é observada a partir de múltiplas poses de câmera, e usa o filtro de Kalman Estendido (EKF) para fazer fusão dessas informações com o IMU. Esses algoritmos não realizam tratamento específico para pontos dinâmicos, mas a presença de sensores adicionais não afetados pelo movimento de outros objetos no ambiente melhora a acurácia em tais ambientes.

Algumas abordagens buscam melhorar o desempenho ao filtrar objetos em movimento. O trabalho de [Kundu, Krishna e Sivaswamy 2009] emprega restrições geométricas e um filtro Bayesiano recursivo para classificar pontos de interesse como dinâmicos ou estáticos. [Migliore et al. 2011] remove pontos inconsistentes de três projeções distintas com base em triangulações. Já o [Nam e Gon-Woo 2020] altera o MSCKF, adicionando um método de rejeição de *outliers* para tratar pontos dinâmicos, e [Fu, Xia e Qiao 2021] utiliza medições de IMU para remover pontos móveis e calcular a odometria com base nos pontos estáticos restantes.

Em uma cena composta por objetos dinâmicos, a diversidade de configurações e relações entre movimento aparente e textura podem criar situações ambíguas e de difícil entendimento visual. Esses problemas são mais frequentes em uma observação local e independente de conhecimento prévio. Redes Neurais de segmentação se destacam pela capacidade de capturar relações entre dados onde a modelagem tradicional é muito complexa de ser realizada.

A segmentação pode ser feita através de redes convolucionais (CNNs), como em [Long, Shelhamer e Darrell 2015] e [Badrinarayanan, Handa e Cipolla 2015]. A U-Net [Ronneberger, Fischer e Brox 2015] introduziu conexões de salto para conectar múltiplas camadas internas do encoder a camadas do decoder, para preservar os detalhes das bordas e os contornos dos objetos. Recentemente, se destaca o uso de transformers [Vaswani et al. 2017] em redes de segmentação. O Vision Transformer [Dosovitskiy et al. 2021] adaptou o transformer, originalmente usado em modelos de linguagem, para apliações em visão. O Swin Transformer [Liu et al. 2021] introduz janelas deslocadas para calculo da atenção, reduzindo a complexidade do transformer de quadrática para linear. O ViT-Adapter [Chen et al. 2022] faz adaptações no Vision Transformer original, como o Injetor de Características Espaciais e Extrator de Características Multi-

escala, para melhorar seu desempenho em detecção e segmentação de objetos. O Lawin Transformer [Yan, Zhang e Wu 2022] introduz um mecanismo de atenção de janela ampla para melhor integrar representações em múltiplas escalas, para otimizar a segmentação em imagens de alta resolução. Há modelos que alcançam melhor desempenho em segmentação ao usar dados multimodais, como o BEiT [Wang et al. 2023]. Também há modelos de larga escala, como o InternImage [Wang et al. 2023], que utilizam de CNNs de para alcançar desempenho próximo ou superior a modelos de transformer. O Segment Anything [Kirillov et al. 2023] foi um grande avanço na tarefa de segmentação, pois é capaz de anotar objetos através de prompts de pontos ou *bounding box*, em *datasets* em que nunca foi treinado.

A segmentação semântica também é aplicada na identificação de objetos que frequentemente estão em movimento. O DynaSlam [Bescos et al. 2018] apresenta um trabalho que usa a Mask-RCNN [He et al. 2017] para segmentar a imagem RGB em classes dinâmicas a priori (pessoa, bicicleta, carro, etc.), e estima a pose da câmera usando a região estática. Também propõe um método geométrico para remover objetos não pertencentes as classes anotadas mas que foram movidos, como um livro carregado por uma pessoa ou uma cadeira sendo empurrada, e combina os resultados do método geométrico com a segmentação. No entanto, a proposta não considera que, apesar da região descartada ser potencialmente móvel, ela pode estar estática no momento e conter textura que podem ser usadas no cálculo do movimento. [Samadzadeh e Nickabadi 2022] combina segmentação semântica e informações geométricas para filtrar pontos dinâmicos, desenvolvendo um algoritmo de SLAM robusto.

Algumas abordadens combinam imagens com fluxo ópticos em redes neurais. [Sevilla-Lara et al. 2016] utiliza o fluxo óptico para aprimorar a segmentação semântica de uma rede DeepLab [Chen et al. 2018]. O SMSNet [Vertens, Valada e Burgard 2017] realiza a segmentação de movimento com base em duas imagens consecutivas usando uma rede convolucional. A rede é treinada inicialmente para segmentação semântica em um dataset urbano, e depois é realizado um segundo treinamento para distinguir quais desses objetos estão em movimento com a adição de dados de fluxo óptico.

Outros trabalhos usam sensores adicionais. O estudo de [Jung et al. 2021] propõe um método iterativo que identifica pontos característicos em movimento com base em informações inerciais e refina a estimativa usando uma rede neural de segmentação.

Os trabalhos que utilizam de métodos geométricos ou sensores inerciais para estimar os objetos em movimento não levam em consideração que se parte de um objeto é dinâmico, o objeto inteiro provavelmente é dinâmico, por tem uma compreensão da imagem menos completa que os métodos de deep learning. Já os trabalhos que utilizaram de redes de segmentação para estimar objetos dinâmicos podem estar limitados a reconhecer apenas os objetos pertencentes a classes pré-definidas no dataset de treino, como

pessoas e veículos. Outra limitação encontrada na literatura é a baixa quantidade de imagens anotadas para segmentação de objetos dinâmicos. [Vertens, Valada e Burgard 2017] treina seu modelo de segmentação em apenas 255 imagens do KITTI-Motion e 2975 do Cityscapes-Motion.

CAPÍTULO 3

Fundamentos teóricos

3.1 Odometria Visual 2D-para-2D

A odometria visual é o processo de estimar a pose de um veículo de forma incremental, examinando as mudanças que o movimento causa na imagem de suas câmeras. Uma de suas vantagens é que não é afetada por derrapamento nas rodas, e permite boa precisão comparado com odometria de rodas. Os algoritmos geralmente presumem que há boa iluminação no ambiente, boa quantidade de texturas e que há mais objetos estáticos que em movimento, e podem ter desempenho piorado caso essas condições não sejam mantidas [Scaramuzza e Fraundorfer 2011].

Ela pode ser monocular (apenas uma câmera), estéreo (duas câmeras) ou pode incluir mais sensores, como de profundidade e inerciais. O pipeline básico da odometria visual monocular é:

- Captura de imagens
- Detecção e descrição de pontos característicos
- Correspondência entre pontos característicos
- Estimativa de movimento
- Otimização local (Bundle adjustment), opcional

3.1.1 Detecção e correspondência de características

A detecção de pontos característicos permite que o sistema encontre pontos adequados para serem rastreados ao longo do tempo. Esses pontos geralmente são estruturas de alto contraste, menos sensíveis a alterações de visualização, idealmente com pouca variância a transformações, usualmente, com a aparência de cantos ou bordas.

Detectores como HARRIS [Harris e Stephens 1988] e FAST (Features from Accelerated Segment Test) [Rosten e Drummond 2005] são opções comuns nos trabalhos da área, baseiam-se no gradiente local e em diferenças estruturadas entre *pixels* vizinhos,

respectivamente, alcançando assim a capacidade de detectar regiões de alto contraste e invariância a luminosidade e transformações Euclidianas[1].

O detector FAST define um circulo de 16 pixeis em volta de um ponto p, como na Figura 3.1 e verifica a presença de conjuntos de pixeis que atendem as condições:

- Conjunto de N pixeis contíguos com intensidade $I_x > I_p + t$
- Conjunto de N pixeis contíguos com intensidade $I_x < I_p - t$

Em que I_x é a intensidade do pixel do círculo, I_p é a intensidade ponto no centro do círculo e t é um valor de limite definido pelo algoritmo. Caso o ponto atenda pelo menos uma das condições, é classificado como um canto.

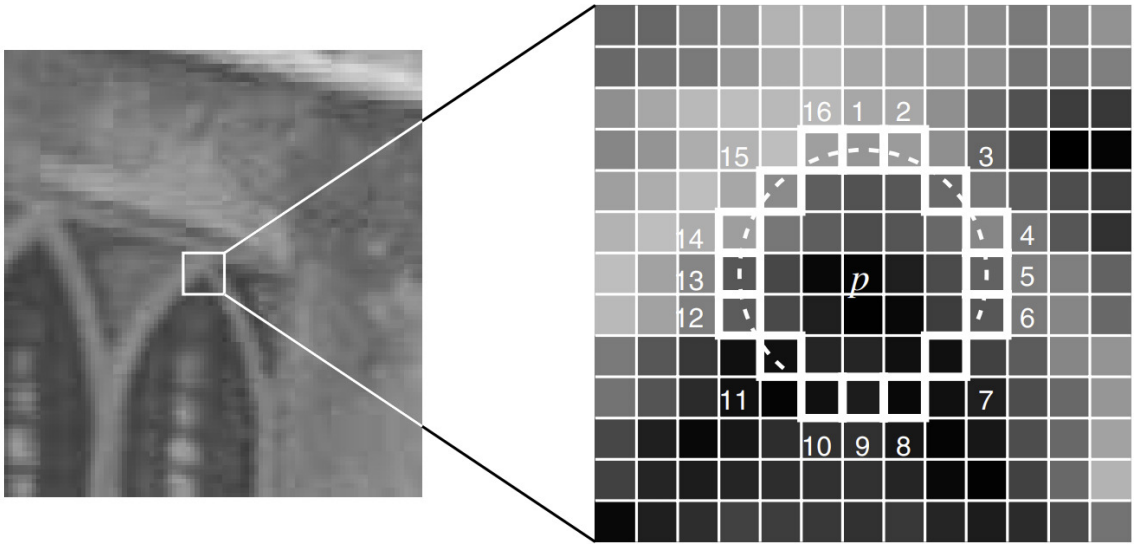


Figura 3.1: Circulo verificado pelo FAST. Fonte: [Rosten e Drummond 2005]

A correspondência entre pontos consiste em identificar uma mesma região em imagens diferentes. Espera-se que, em pequenas alterações nas condições de visualização, a mesma região compartilhe uma mesma aparência visual. Para isso, um algoritmo de correspondência deve considerar uma função de similaridade que permita comparar a representação de regiões entre si.

Um ponto de interesse, ou uma característica, representa uma região local em uma posição específica na imagem. Sendo assim, a representação de um ponto/característica considera a sua vizinhança local diretamente ou uma descrição desta. Em uma comparação direta *pixel*-a-*pixel*, é comum utilizar métricas que consideram as intensidades dos *pixels* adjacentes, tais como a Soma das Diferença Absolutas (SDA), Correlação Cruzada Normalizada (CCN) e a Distância Euclidiana [Li 2017]. Por outro

[1]Uma transformação Euclidiana é o conjunto de transformações lineares com o potencial de alterar posição e orientação de objetos em um espaço Euclidiano, ou seja, somente as transformações de rotação e translação são realizadas [Szeliski 2010].

lado, a avaliação direta das intensidades dos *pixels* torna a métrica de similaridade muito sensível a distorções de visualização da cena, tanto em pose e intensidade luminosa, sendo necessária uma descrição mais elaborada da região de interesse.

A descrição de pontos é tema de múltiplos trabalhos da área de correspondência de características. Ela consiste em estabelecer uma assinatura representativa do ponto em questão, usualmente uma sequência numérica, que permita a comparação e o cálculo de similaridade entre múltiplos pontos de modo invariante a distorções de visualização [Chien et al. 2016]. Entre os algoritmos de descrição estão o BRIEF (Binary Robust Independent Elementary Features) [Calonder et al. 2010] e o BRISK (Binary Robust Invariant Scalable Key- points) [Leutenegger, Chli e Siegwart 2011].

O BRIEF realiza um teste binário de intensidade em locais de pixel escolhidos aleatoriamente ou com base em padrões (como a distribuição gaussiana) em uma caixa suavizada em torno do ponto-chave. O descritor resultante é uma sequência de valores binários (geralmente de 256 bits). A expressão geral do BRIEF é dada por:

$$D = \sum_{i=1}^{n} 2^{i-1} \cdot d(p_i, q_i) \tag{3-1}$$

onde d(p, q) é uma função binária definida como:

$$d(p,q) = \begin{cases} 1 & \text{se} I(p) < I(q) \\ 0 & \text{caso contrário} \end{cases} \tag{3-2}$$

em que I(p) denota a intensidade no ponto p.

Alguns algoritmos realizam ambas a detecção e descrição de pontos característicos, como o *Scale-Invariant Feature Transform* (SIFT) [Lowe 1999] e o *Speeded Up Robust Features* (SURF) [Bay, Tuytelaars e Gool 2006], amplamente empregados pela capacidade de lidar com problemas de escala. Ambos os detectores realizam a detecção no domínio espaço-escala a partir da análise multi-resolução de imagem usando Pirâmides Laplacianas.

As características detectadas pelo SURF são os locais em que a Hessiana da imagem tem um máximo local. O algoritmo emprega um conjunto de otimizações, como o uso de imagens integrais para calcular as convoluções e a aproximação da Hessiana baseado em filtros de caixa. Já o seu descritor é calculado a partir de uma janela de 20x20 pixels em torno do ponto de interesse, subdividida em 4x4 sub-janelas de 5x5 pixels. Para cada sub-janela, o SURF aplica a transformada Wavelet Haar em 4 direções e as concatena em um vetor de 64 elementos. Em seguida é feito a soma dos vetores das sub-janelas para obter o descritor final.

O ORB (Oriented FAST and Rotated BRIEF) [Rublee et al. 2011] gera pirâmides de múltiplas escalas da imagem e aplica o FAST em cada uma, para detectar as featu-

res de forma invariante a escala. Em seguida aplica o rBRIEF, uma modificação do BRIEF invariante a rotação, para gerar os descritores. A Figura 3.2 mostra uma correspondência encontrada usando o orb.

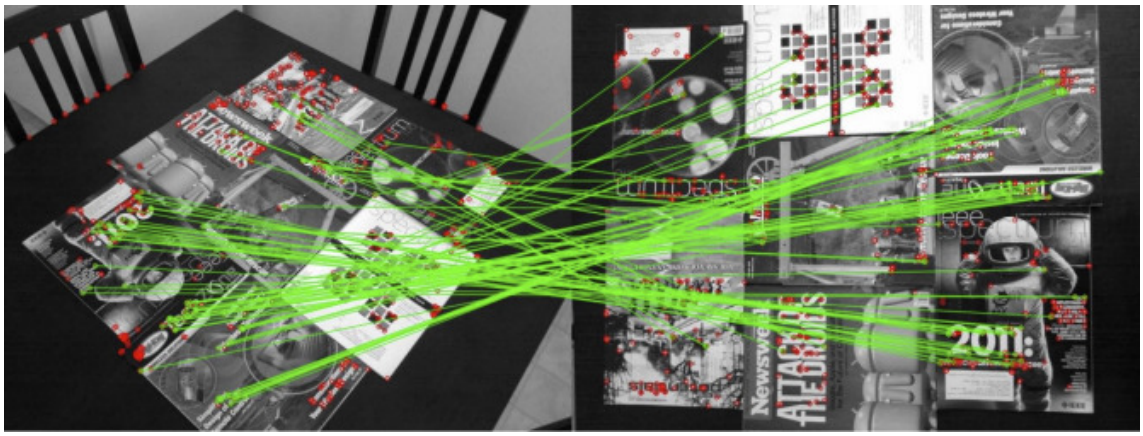


Figura 3.2: Correspondência por ORB. Fonte: [Rublee et al. 2011]

Para obter a correspondência entre os pontos característicos já descritos, podem ser usados algoritmos como o KNN (K-Nearest Neighbors) e o FLANN (Fast Library for Approximate Nearest Neighbors) [Muja e Lowe 2009]. O SuperGlue [Sarlin et al. 2020] é uma alternativa que faz a correspondência através de uma rede neural de grafos com atenção, e consegue desempenho bom mesmo em ambientes desafiadores.

3.1.2 Estimativa de movimento

A estimativa de movimento consiste em estimar a translação e rotação entre a imagem anterior e a atual, a partir dos pontos correspondentes estimados na etapa anterior. As translações e rotações entre imagens sequencias são concatenadas para recuperar o trajeto completo. Há três formas de estimar o deslocamento:

- 2D-para-2D: Os pontos anteriores f_{k-1} e atuais f_k são definidos por coordenadas 2D
- 3D-para-2D: Os pontos anteriores f_{k-1} são 3D, e os pontos atuais f_k são a projeção 2D dos pontos anteriores na imagem atual I_k
- 3D-para-3D: Os pontos anteriores f_{k-1} e atuais f_k são definidos por coordenadas 3D. Este método é mais comum em câmeras estéreo, pois os pontos 3D podem ser triangulados a partir dos pontos 2D das imagens da esquerda e direita.

Serão definidos as estimativas 2D-para-2D e 3D-para-2D a seguir. A estimativa 3D-para-3D não é estudada neste trabalho por não ser usada em sistemas de única câmera monocular.

Estimativa de movimento 2D-para-2D

Quando a câmera observa um mesmo ponto X em duas poses diferentes, há um plano epipolar que passa pelos dois pontos p e p' e pela origem do sistema de coordenadas da câmera C_k e C_{k-1}, como é visto na Figura 3.3.

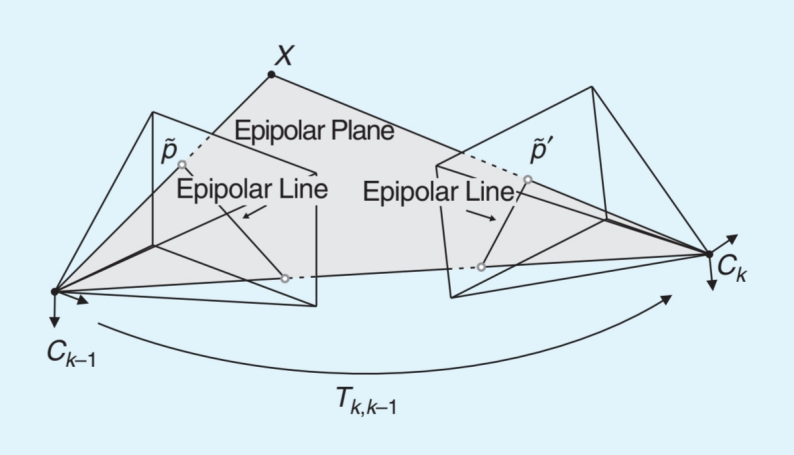


Figura 3.3: Geometria Epipolar. Fonte: [Scaramuzza e Fraundorfer 2011]

Pode ser definida a matriz fundamental, relaciona os pontos característicos correspondentes de duas imagens:

$$p_k^T F p_{k-1} = 0 \tag{3-3}$$

onde p_k e p_{k-1} são os pontos característicos da imagem k e $k-1$ respectivamente. O termo $F \cdot p_{k-1}$ descreve a linha em que o ponto correspondente p_k é encontrado. Essa linha é chamada de linha epipolar.

Através da matriz fundamental é possível calcular a matriz essencial, que relaciona pontos no sistema de coordenadas do primeiro *frame* com pontos no sistema de coordenadas do segundo *frame*. A matriz essencial é dada por:

$$E = (K)^T F K \tag{3-4}$$

Em que K é a matriz de câmera, que relaciona os pontos do mundo com os pontos da imagem. A matriz K é dada por:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3-5}$$

onde f_x e f_y são as distâncias focais da câmera, e c_x e c_y são as coordenadas do centro da imagem. Estes valores podem ser obtidos a partir da calibração da câmera.

A partir dos pontos correspondes e matriz de câmera, a matriz essencial pode ser estimada usando o algoritmo de cinco pontos [Nister 2004].

A matriz essencial pode ser decomposta em matrizes R e t, que são as matrizes de rotação e de translação entre uma imagem e a próxima:

$$E = \lfloor t \rfloor_x R \tag{3-6}$$

onde $[t]_x$ é a matriz anti-simétrica de t. A decomposição pode ser feita aplicando decomposição de valores singulares (SVD).

Com a rotação e translação entre duas imagens, a posição e orientação da câmera é estimada de forma iterativa. A orientação da câmera na imagem k é dada por:

$$R_k = RR_{k-1} \tag{3-7}$$

onde R_{k-1} é a orientação da câmera na imagem $k-1$. A posição da câmera na imagem k é dada por:

$$t_k = t_{k-1} + \lambda \cdot R_{k-1} t \tag{3-8}$$

onde t_{k-1} é a posição da câmera na imagem $k-1$. t é um vetor unitário que aponta na direção da translação entre as imagens. É necessário aplicar uma escala λ para obter a posição da câmera em metros. No caso de uma câmera monocular, a escala é desconhecida, e precisa ser estimada. A escala pode ser estimada usando sensores externos. O VINS-Mono [Qin, Li e Shen 2018], por exemplo, usa um IMU para estimar a escala.

3.2 Odometria Visual 3D-para-2D

A odometria 3D para 2D consiste em encontrar a transformação que projeta pontos 3D em uma imagem 2D, como mostra a figura 3.4. A transformação $T+k$ é obtida através da correspondência entre os pontos 3D X_{k-1} e os pontos 2D p_k.

A solução é dada pela transformação T_k que reduz o erro de reprojeção:

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = argmin_{T_k} \sum_i ||p_k^i - p_{k-1}^i||^2 \tag{3-9}$$

Em que p_k^i é a reprojeção dos pontos 3D X_{k-1}^i na imagem I_k de acordo com a transformação T_k. Este problema é chamado de camera resection ou PnP (perspective from n points).

Uma solução para o problema PnP é o algoritmo DTL (direct linear transformation) [Hartley e Zisserman 2003]. Uma correspondência 3D-para-2D resultam nas restri-

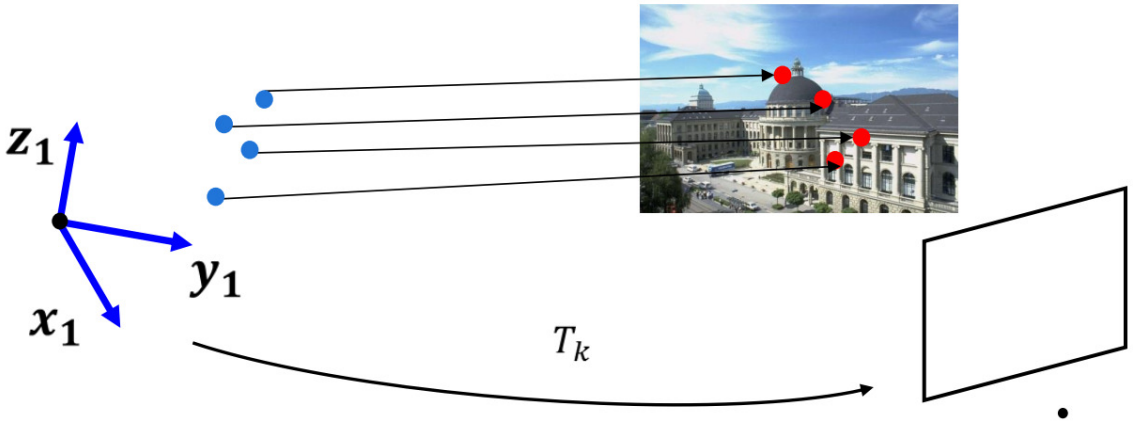


Figura 3.4: Projeção de pontos. Fonte: [Scaramuzza e Fraundorfer 2011]

ções:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & -x & -y & -z & -1 & x\widetilde{v} & y\widetilde{v} & z\widetilde{v} & \widetilde{v} \\ x & y & z & 1 & 0 & 0 & 0 & 0 & -x\widetilde{u} & -y\widetilde{u} & -z\widetilde{u} & -\widetilde{u} \end{bmatrix} \begin{bmatrix} P^1 \\ P^2 \\ P^3 \end{bmatrix} = 0 \tag{3-10}$$

Em que P^j é a linha j do vetor de projeção $P = [R|t]$, x, y e z são coordenadas do ponto 3D X_{k-1} e $\widetilde{u}$ e $\widetilde{v}$ são as coordenadas do ponto 2D p_k. A partir de pelo menos seis correspondências de pontos pode ser formados um sistema de equações lineares na forma AP=0, e P pode ser calculado utilizando SVD. Decomponto P_k é obtido as matrizes de translação e rotação t e R.

Para imagens monoculares, os pontos devem ser triangulados a partir de duas imagens anteriores adjacentes (I_{k-2} e I_{k-1}), e o problema PnP é resolvido para a projeção na imagem atual I_k. Ao obter R e t, a estimativa é atualizada pelas equações 3-7 e 3-8. De forma iterativa, mais pontos são triangulados, adicionados a nuvem de pontos e projetados na próxima imagem.

3.2.1 Local Optimization (Bundle Adjustment)

A odometria é calculada concatenando as transformações estimadas *frame-a-frame*, mas é possível orimizar a estimativa utilizando uma janela de *frames* anteriores. As poses calculadas podem ser representadas por um grafo de poses, em que as poses de câmera são os nós e as transformações entre poses são as arestas entre os nós. As restrições de arestas e_{jj} definem a função de custo:

$$\sum_{e_{ij}} ||C_i - T_{e_{ij}}||^2 \tag{3-11}$$

Em que $T_{e_{ij}}$ é a transformação entre as poses j e j e C_i é a i-ésima pose da câmera. O método *Pose-Graph Optimization* [Olson, Leonard e Teller 2006] estima as poses de câmera que minimizam essa função de custo, utilizando um método de otimização não-linear como o Levenberg-Maquardt [Levenberg 1944].

O Loop Closure é um caso especial de otimização de grafos, em que pontos 3D são observados novamente após um longo tempo sem serem observados. As restrições de loop podem ser encontradas ao avaliar a similaridade entre imagens atuais e imagens anteriores. Um método é representar a imagem como um contunto de palavras visuais (*Bag of Words*) e calcular a distancia entre as representações de duas imagens.

O método de Bundle adjustment [Triggs et al. 2000] é semelhante ao *Pose-Graph Optimization*, mas também otimiza os pontos 3D do mapa. No bundle adjustment, a função de erro a ser minimizada é a de erro de reprojeção:

$$argmin_{X^i,C_k} \sum_{i,k} ||p_i^k - g(X^i, C_k)||^2 \tag{3-12}$$

Em que p_i^k é a projeção 2D do i-ésimo ponto 3D X^i observado na k-ésima imagem, e $g(X^i, C_k)$ é a reprojeção do ponto 3D de acordo com a pose C_k da imagem. Essa função de erro também é minimizada por métodos de otimização não-linear como o Levenberg-Maquardt. É necessário uma inicialização das poses e pontos 3D, que geralmente é providenciado pelo pipeline básico de odometria.

O bundle adjustment é capaz de reduzir erros de estimativa por usar medições provenientes de mais de dois *frames*. O número de *frames* otimizadas depende principalmente dos recursos computacionais disponíveis.

3.2.2 Implementações de odometria visual

As duas implementações de odometria visual estudadas neste trabalho é o pyslam [Freda 2024] e a versão monocular do ORB-SLAM2 [Mur-Artal e Tardós 2017]. Ambos são algoritmos 3D-para-2D, como definido na seção 3.2. Ambos fazem uma etapa de inicialização calculando a matriz de homografia, como definido na seção 3.1.2. Os pontos são triangulados e utilizados para estimar o movimento pela redução do erro de reprojeção, com otimização por Bundle Adjustment (seção 3.2.1).

O ORB-SLAM2 utiliza de *Bag-of-Words* para encontrar locais já observados anteriormente, para realizar Loop Closure e para re-localizar caso haja uma falha de *tracking*.

Os dois algoritmos também realizam seleção e descarte de *Keyframes*, processo que descarta imagens capturadas até que uma imagem, a *keyframe*, resulte na redução de incerteza dos pontos 3D triangulados.

3.3 Segmentação Semântica

A segmentação semântica consiste em atribuir uma categoria semântica a cada pixel de uma imagem, agrupando assim os pixels que pertencem ao mesmo objeto ou região. O objetivo é gerar uma compreensão mais detalhada do conteúdo visual, identificando o que há em uma imagem e onde esses objetos estão localizados [Csurka, Volpi e Chidlovskii 2023]. Um exemplo de resultado de segmentação semântica pode ser vista na Figura 3.5

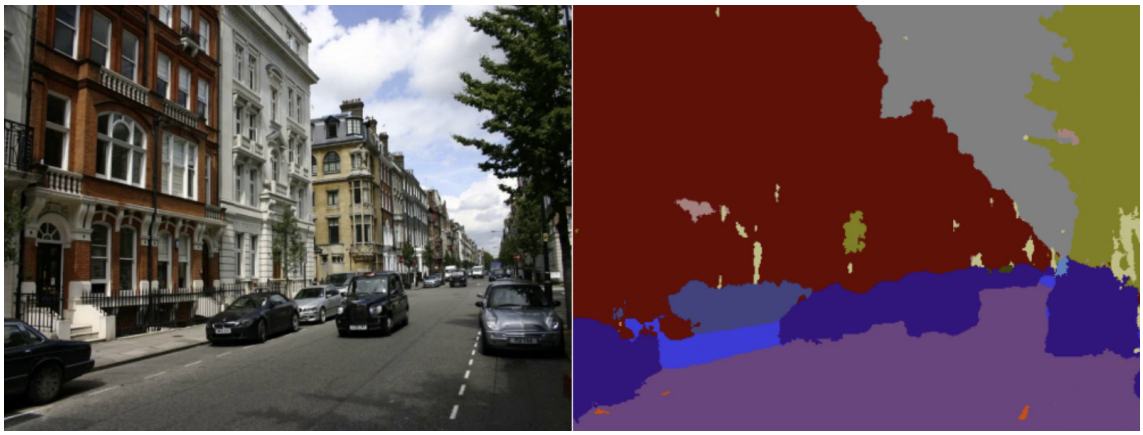


Figura 3.5: Resultado da SegNet. Fonte: [Badrinarayanan, Handa e Cipolla 2015]

Tipicamente, esse problema é abordado utilizado métodos aprendizado supervisionado, utilizando imagens anotadas pixel a pixel para treinar métodos de deep learning. Em geral, os métodos buscam minimizar a entropia cruzada entre os píxeis preditos e os píxeis de ground truth.

Métodos de deep learning obteram grande sucesso nesta área, mas dependem da disponibilidade de grandes quantidades de imagens anotadas. Anotar cada imagem pixel a pixel é um processo muito demorado e custoso, e por isso alguns métodos dados simulados para produzir uma grande quantidade de imagens diversas [Csurka, Volpi e Chidlovskii 2023].

Existem vários modelos e abordagens para realizar a segmentação semântica, sendo alguns deles:

Redes Neurais Convolucionais (CNNs): A segmentação pode ser realizada por redes puramente convolucionais, como o artigo [Long, Shelhamer e Darrell 2015], que modifica a AlexNet [Krizhevsky, Sutskever e Hinton 2012], originalmente uma rede de classificação, para realizar segmentação.

SegNet [Badrinarayanan, Handa e Cipolla 2015]: SegNet é outra arquitetura de segmentação semântica baseada em CNNs. Essa rede utiliza uma estrutura de codificador-

decodificador. O codificador captura informações contextuais da imagem, enquanto o decodificador reconstrói essas informações a partir das baixas dimensões do espaço latente do codificador. A SegNet utiliza técnicas como o up-sampling e a passagem de informações de índice de pooling do codificador para o decodificador para gerar os mapas semânticos de mesma resolução da imagem de entrada.

U-Net [Ronneberger, Fischer e Brox 2015]: A U-Net também adota uma estrutura de codificador-decodificador com uma forma simétrica de "U". Diferentemente da SegNet, a U-Net utiliza conexões de salto, onde as informações de camadas do codificador são concatenadas às camadas correspondentes do decodificador, permitindo transmissão eficiente de informações de alta resolução. Essa abordagem é especialmente útil para tarefas de segmentação em imagens médicas, onde a precisão na localização de bordas é crucial. A arquitetura da Unet é detalhada na Figura 3.6.

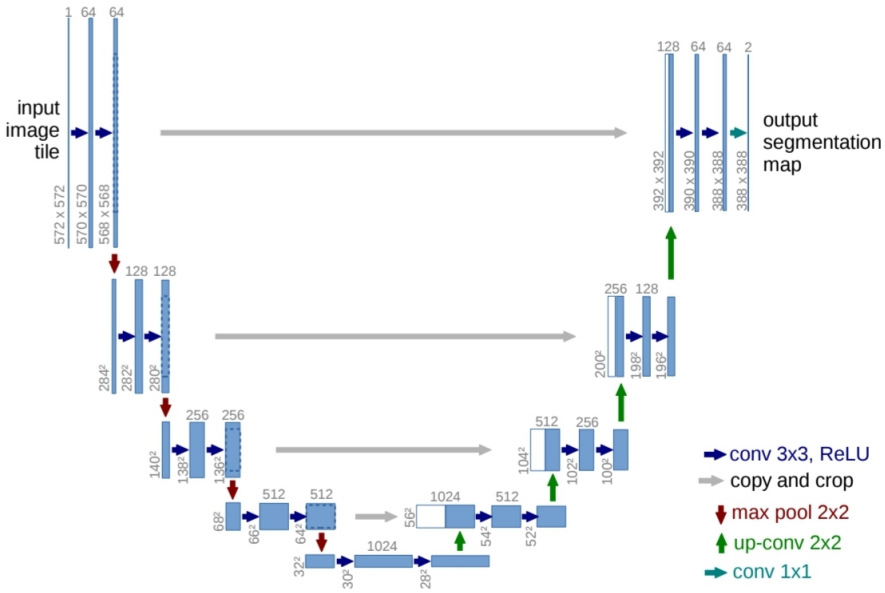


Figura 3.6: Arquitetura da Unet. Fonte: [Ronneberger, Fischer e Brox 2015]

Mask R-CNN [He et al. 2017]: O Mask R-CNN é um modelo de detecção e segmentação de instâncias que estende o Faster R-CNN [Ren et al. 2015], um modelo de detecção de objetos, adicionando uma nova componente que permite ao modelo gerar uma máscara binária para cada instância de objeto detectada, juntamente com seus rótulos de classe e bounding box.

Lawin Transformer [Yan, Zhang e Wu 2022]: Visa melhorar o desempenho de segmentação semântica ao integrar representações em múltiplas escalas usando um mecanismo de atenção de janela ampla (large window attention). O Lawin Transformer busca otimizar a segmentação semântica, especialmente em imagens de alta resolução,

utilizando uma abordagem que seja computacionalmente eficiente e capaz de entender o contexto em diferentes escalas. Ele foca em reduzir o custo computacional, um problema recorrente nos Transformers voltados para segmentação semântica, ao mesmo tempo em que tenta manter ou melhorar a precisão em benchmarks conhecidos, como Cityscapes, ADE20K e COCO-Stuff.

Sua arquitetura pode ser vista na figura 3.7.

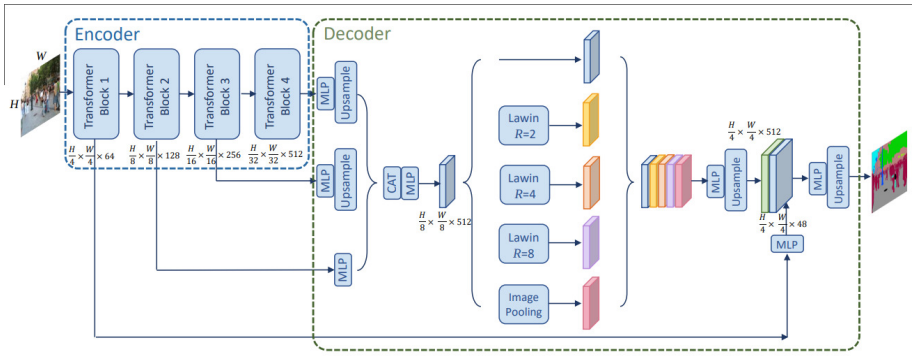


Figura 3.7: Arquitetura do Larwin Transformer

O Lawin Transformer possui uma arquitetura composta por:

- Encoder Hierárquico de Visão Transformer (HVT): Utiliza um Transformer hierárquico, como o Swin Transformer e MiT, para construir uma representação inicial das características visuais.
- LawinASPP (Large Window Attention Spatial Pyramid Pooling): Um módulo no decoder que usa a atenção de janela ampla para capturar contextos em múltiplas escalas. Esse módulo utiliza o mecanismo de atenção de janela ampla para permitir que cada patch de consulta (query patch) observe uma área maior ao seu redor, aproveitando uma pirâmide de contextos em várias escalas.

ViT-Adapter [Chen et al. 2022]: É uma adaptação do Vision Transformer (ViT) [Dosovitskiy et al. 2021] para tarefas de predição densa, como detecção e segmentação de objetos. O ViT, originalmente desenvolvido para NLP, possui uma arquitetura geral que carece de suposições específicas para visão, o que o torna menos eficaz em tarefas densas como detecção de objetos e segmentação. Para resolver isso, o ViT-Adapter introduz adaptações para permitir que o ViT execute comparativamente bem em tarefas visuais específicas sem necessidade de pré-treinamento. Sua arquitetura é apresentada na imagem 3.8.

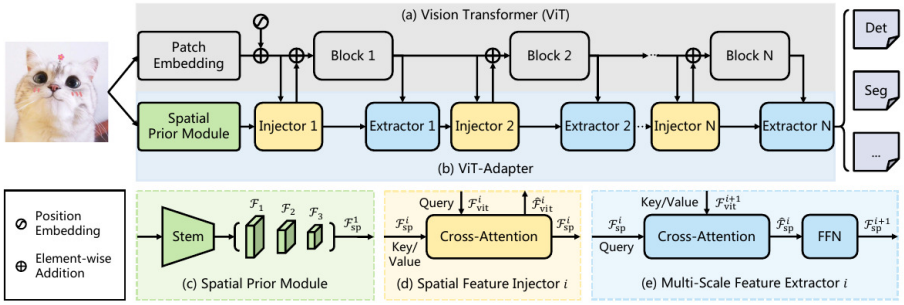


Figura 3.8: Arquitetura do ViT-Adapter

A arquitetura tem duas partes principais:

- ViT Base: Um Vision Transformer simples que divide a imagem em patches (16x16), transformando-os em tokens de alta dimensionalidade. Esses tokens passam por camadas de codificação.
- ViT-Adapter: Um módulo adicional que introduz vieses espaciais, específicos para visão, ao ViT. Ele possui três componentes:
  - Módulo de Prior Espacial: Aplica convoluções para extrair informações locais e espaciais da imagem.
  - Injetor de Características Espaciais: Utiliza atenção cruzada para integrar as características espaciais na ViT.
  - Extrator de Características Multiescala: Constrói uma pirâmide de características multiescala necessárias para predições densas.

Essa abordagem permite que a ViT atinja desempenhos comparáveis ou superiores aos transformers específicos para visão, especialmente ao aproveitar o pré-treinamento multi-modal.

InternImage [Wang et al. 2023]: Rede convolucional (CNN) de grande escala que busca obter desempenho melhor com o aumento na quantidade de dados no grande de parâmetros, assim como os transformers visuais (VITs). Ela incorpora convoluções deformáveis (DCNv3) como operador central, permitindo uma agregação espacial adaptativa e dependências de longo alcance, características presentes nos ViTs. Isso reduz o viés indutivo tradicional das CNNs, tornando-as mais robustas e eficazes na aprendizagem de padrões complexos.

Os blocos de convoluções deformáveis são organizados em uma estrutura hierárquica, similar às redes transformers, incluindo camadas de normalização e uma rede feed-forward. A arquitetura adota um operador convolucional deformável de tamanho 3x3 que adapta o campo receptivo dinamicamente, proporcionando eficiência de memória e computacional. Os modelos da InternImage variam de 30 milhões a 1 bilhão de

parametros, alcançando desempenho comparável aos modelos transformers recentes.

BEIT-3 [Wang et al. 2023]: É uma rede neural multimodal desenvolvida para tarefas de visão e linguagem. O objetivo central do BEIT-3 é consolidar o aprendizado em tarefas de visão, linguagem e visão-linguagem usando uma única arquitetura de Transformer, projetada para lidar com dados de múltiplas modalidades. A arquitetura do BEIT-3 baseia-se em "Multiway Transformers", que compartilham módulos de atenção e integram diferentes redes feed-forward específicas para cada modalidade (visão, linguagem e visão-linguagem). Essa abordagem permite que o modelo realize tanto codificação específica de modalidade quanto fusão profunda entre as modalidades.

O BEIT-3 realiza um pré-treinamento com a tarefa de "masked data modeling", aplicando-a a dados monomodais (imagens ou textos) e multimodais (pares imagem-texto). Neste processo, o modelo é treinado para prever textittokens ausentes em textos ou *patches* em imagens, considerando imagens como uma "língua estrangeira"(ImgLish), o que facilita o uso de imagens e textos de forma integrada e sem distinções fundamentais de modelagem. A escala e a modularidade do BEIT-3 permitem seu ajuste fino para diversas tarefas, incluindo classificação de imagens, segmentação semântica, resposta a perguntas visuais e geração de legendas.

3.3.1 Segment Anything

O Segment Anything [Kirillov et al. 2023] é um modelo de segmentação com o objetivo de permitir segmentar por *prompts*, como pontos e *bounding boxes*, sendo é capaz de transferir de forma *zero-shot* para tarefas e *datasets* em que não foi treinado.

A arquitetura do segment anything pode ser visto na figura 3.9.

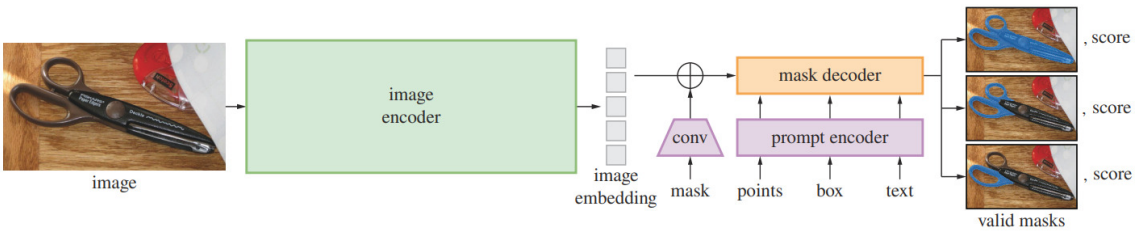


Figura 3.9: Arquitetura do Segment Anything

Os componentes de sua arquitetura são:

Encoder de imagem: Utiliza um transformer visual chamado masked encoder [He et al. 2022] pré-treinado, modificado para processar imagens de alta resolução. Ele pode ser aplicado uma vez para cada imagem antes de inciar o processo de *prompt*, para reduzir o tempo de inferência ao testar multiplos prompts.

Encoder de prompt: Recebe como entrada prompts esparsos (pontos, *bounding box* ou texto) ou denso (máscara de segmentação). Pontos e *bounding box* é representado por *positional encoding* somado a *embeddings* aprendidos para cada tipo de entrada, e texto é representado por um encoder do CLIP [Radford et al. 2021]. As *embeddings* de máscaras são obtidas por convoluções e somadas ao *embeddings* da imagem.

Decoder de máscara: Responsável por calcular a máscara a partir dos *embeddings* de imagem e de prompt e de um token de saída. Utiliza um bloco de decoder transformer seguido de uma cabeça de predição de máscaras. O decoder transformer usa self-attention e cross-attention nas direções prompt-para-imagem e e imagem-para-prompt para atualizar os *embeddings*. A resolução das *embeddings* da imagem são aumentadas, e é aplicado uma MLP seguido de um classificador dinâmico linear ao token de saída, para obter as probabilidades de máscara a cada pixel da imagem. Para lidar com ambiguidades, são geradas três máscaras de saídas, com a confiança em cada uma.

O treinamento do modelo foi feito em três estágios:

Anotação manual auxiliada: O SAM foi treinado em datasets públicos de segmentação, e depois foi utilizado para auxiliar anotadores no processo de anotação de 120 mil imagens adicionais.

Estágio semi-automático: Imagens foram inicialmente anotadas pelo SAM, e os anotadores anotaram objetos adicionais que o SAM não segmentou, com o objetivo de aumentar a variedade de objetos segmentados. O número de imagens aumentada aumentou para 180 mil, com 44 a 72 objetos anotados por imagem.

Estágio automático: Foram geradas máscaras com um prompt de grade de 32x32 pontos, usando as três saídas do modelo (que geralmente são subparte, parte e objeto inteiro). Foram escolhidos as máscaras com mais confiança e com maior estabilidade ao alterar o *threshold* de probabilidade, filtrando duplicados. Esse processo é aplicado para todas as 1.1 bilhão de imagens do dataset, resultando no dataset SA-1B com 11 milhões de máscaras.

A figura 3.10 mostra exemplos do SAM testado em múltiplos datasets, em que o prompt foi um único ponto em cada objeto. Cada cor representa um objeto segmentado. O SAM foi capaz de segmentar uma grande variedade de objetos em uma grande quantidade de datasets disponíveis na literatura.

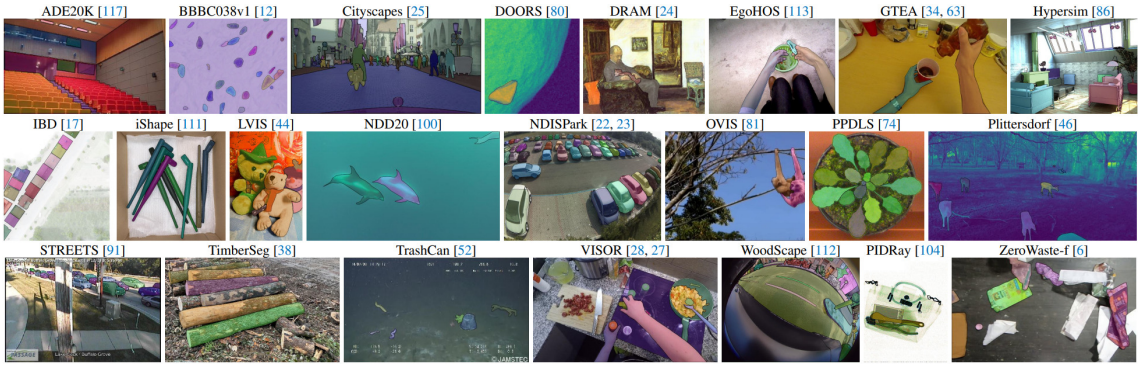


Figura 3.10: Segment Anything testado em múltiplos datasets. Fonte: [Kirillov et al. 2023]

3.4 Fluxo Óptico

O Fluxo óptico é um campo denso de vetores de deslocamento, e representa o movimento aparente de pixeis em imagens adjacentes. Por ser causado pelo movimento entre objetos e a câmera, ele pode dar informações importantes sobre a localização dos objetos e como eles se movimentam. Suas descontinuidades podem ajudar a segmentar imagens em regiões correspondentes a objetos diferentes [Horn e Schunck 1981].

Métodos de fluxo ótico presumem que o brilho é constante, de acordo com a equação:

$$I(x, y, t) = I(x + \Delta X, y + \Delta Y, t + \Delta T) \tag{3-13}$$

Em que $I(x, y, t)$ é a intensidade de cada pixel, x e y são as coordenadas do pixel, e t é o tempo. Ou seja, cada píxel (x, y) se move para $(x + \Delta x, y + \Delta y)$ em um intervalo de tempo ΔT. Desde que Δx, Δy e Δt sejam pequenos, podemos aproximar a equação acima através de uma expansão de Taylor:

$$I(x + \Delta X, y + \Delta Y, t + \Delta T) = I(x, y, t) + \Delta X \frac{\partial I}{\partial x} + \Delta Y \frac{\partial I}{\partial y} + \Delta T \frac{\partial I}{\partial t} + (\text{termos de ordem superior}) \tag{3-14}$$

Removendo termos de ordem superior para linearizar:

$$\frac{\partial I}{\partial x}\Delta X + \frac{\partial I}{\partial y}\Delta Y + \frac{\partial I}{\partial t}\Delta t = 0 \tag{3-15}$$

Dividindo por Δt para obter as velocidades V_x e V_y:

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0 \tag{3-16}$$

Que pode ser simplificada para:

$$\nabla I \cdot V = -I_t \tag{3-17}$$

Em que I_t é a derivada parcial de I em relação a t. Por ter duas incógnitas, a equação 3-17 é indeterminada, necessitando de mais uma restrição para ser resolvida. Lucas-Kanade [Lucas e Kanade 1981] propõe a restrição de que o fluxo ótico seja constante em uma vizinhança de pixels. Para n pixels, as equações são:

$$\begin{aligned} \nabla I_1 \cdot V &= -I_{1t} \\ \nabla I_2 \cdot V &= -I_{2t} \\ &\vdots \\ \nabla I_n \cdot V &= -I_{nt} \end{aligned}$$

Ou, em forma matricial:

$$A \cdot v = b \tag{3-18}$$

Em que A é a matriz Jacobiana (derivadas de I em relação a x e y), v é o vetor de velocidades $[V_x, V_y]^T$ e b é o vetor de intensidades $[-I_{t1}, -I_{t2}, \ldots, -I_{tn}]^T$. Lucas-Kanade obtém uma solução aproximada através de mínimos quadrados, resolvendo o sistema:

$$A^T A v = A^T b \tag{3-19}$$

Além de métodos clássicos como o de Lucas-Kanade, há trabalhos mais recentes que utilizam redes neurais. O DICL [Wang et al. 2020] visa solucionar os desafios de escalabilidade e precisão no cálculo de fluxo óptico usando redes neurais. Sua abordagem busca evitar o alto custo computacional associado à construção de volumes de características 5D e convoluções em 4D, necessário para combinar deslocamentos em 2D com altura, largura e dimensões de características. Em vez disso, o DICL propõe uma série de inovações que reduzem a dimensionalidade e aumentam a precisão da correspondência entre pixels. Sua arquitetura pode ser vista na figura 3.11.

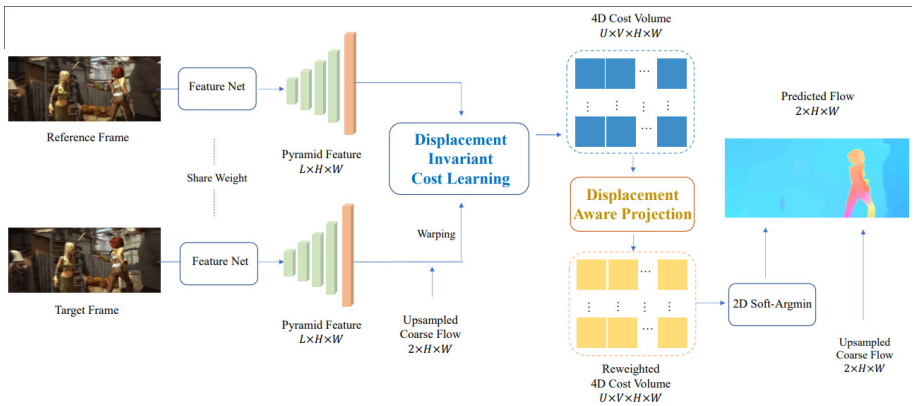


Figura 3.11: Arquitetura do DICL. Fonte: [Wang et al. 2020]

Os componentes da arquitetura são:

- **Feature Net**: Extrai características do par de imagens de entrada em diferentes níveis de resolução, através de convoluções. Ela gera uma pirâmide de características, com 1/4, 1/8, 1/16, 1/32, e 1/64 da resolução total. Isso ajuda o modelo a capturar detalhes tanto globais quanto locais, melhorando a precisão na estimação de movimento em regiões de diferentes tamanhos e texturas.
- **Displacement-Invariant Cost Learning (DICL)**: Em métodos tradicionais, a construção de um volume de características para o fluxo óptico requereria um volume de 5D, o que é impraticável em termos de consumo de memória e processamento. O DICL resolve isso aplicando convoluções em 2D para cada deslocamento individual, formando um volume de custos em 4D em vez de 5D. Para cada deslocamento candidato, o DICL cria um mapa de características concatenando as características do pixel na imagem de referência com as características do pixel deslocado na imagem alvo. Essas características concatenadas são processadas por uma rede de correspondência, que é a mesma para cada deslocamento, e que aplica convoluções 2D para gerar o custo de correspondência para cada hipótese de deslocamento.
- **Displacement-Aware Projection Layer (DAP)**: Reescala os valores no volume de custo com base nas correlações entre diferentes hipóteses de deslocamento. O DAP ajusta o volume de custo ao reponderar os custos em cada plano de deslocamento (dimensão do deslocamento), considerando deslocamentos próximos e ajustando os valores para reduzir os efeitos de distribuições multimodais.
- **2D Soft-Argmin**: Projeta o volume de custo resultante em uma estimativa final de fluxo óptico, permitindo obter a direção e magnitude do movimento para cada pixel. O soft-argmin funciona ao aplicar uma função softmax ao longo da dimensão de deslocamento, atribuindo uma probabilidade a cada possível deslocamento e, então, calculando a média ponderada dessas probabilidades para obter a estimativa final.

O modelo foi inicialmente pré-treinado no dataset sintético FlyingChairs [Dosovitskiy et al. 2015], que contém pares de imagens com movimentos artificiais, permitindo uma ampla variedade de deslocamentos e situações para o aprendizado inicial. A seguir, o modelo foi treinado no dataset FlyingThings [Mayer et al. 2016], um conjunto de dados maior e mais complexo que simula cenários 3D realistas. Esta etapa é crucial para o modelo aprender correspondências mais detalhadas e complexas em cenas mais variadas. Para adaptar o modelo a condições específicas, foi feito o ajuste fino em datasets especializados, como o Sintel (para fluxo em animações complexas) e KITTI (para cenários de direção autônoma).

3.5 Rastreamento de Múltiplos Objetos em Vídeo

O problema de Rastreamento de Múltiplos Objetos, ou *Multiple Object Tracking (MOT)* é o desafio de associar objetos detectados em diferentes quadros de uma sequência de vídeo, assegurando que cada objeto seja corretamente identificado e rastreado ao longo do tempo. Essa tarefa envolve a associação de dados para conectar detecções de objetos entre quadros consecutivos, essencial para o acompanhamento contínuo de cada objeto na cena. Para alcançar essa associação, métodos de rastreamento utilizam modelos de movimento e de aparência dos objetos [Bewley et al. 2016].

O SORT [Bewley et al. 2016] é método simples para resolver este problema. O Filtro de Kalman é usado para modelar e prever a posição dos objetos de um quadro para o próximo. O estado de cada objeto é representado por um vetor que inclui:

- Posição (u, v) - localização no plano da imagem.
- Escala (s) - tamanho da área delimitada.
- Razão de aspecto (r) - proporção entre largura e altura do objeto.
- Velocidade nas direções horizontal e vertical (u, v) e na escala (s).

O SORT presume um movimento com velocidade constante, o que permite predizer a posição e tamanho dos objetos no quadro seguinte, mesmo na ausência de detecção direta (como em casos de oclusão curta).

Para associar detecções do quadro atual com objetos rastreados anteriormente, é necessário resolver um problema de associação de dados. O SORT utiliza o Algoritmo Húngaro para otimizar o custo de associação, calculado pela métrica de Intersection over Union (IoU) entre as caixas delimitadoras previstas e as detecções atuais. Esse processo assegura que cada detecção seja atribuída ao objeto rastreado mais provável, respeitando um limite mínimo de IoU para evitar associações imprecisas.

Também há métodos que utilizam de redes neurais para melhor entender a aparência dos objetos rastreados. O BoT-SORT [Aharon, Orfaig e Bobrovsky 2022] busca melhorar a precisão de rastreamento, especialmente em cenários de alta complexidade, como vídeos com movimentos de câmera e cenas com muitas sobreposições. Sua arquitetura pode ser vista na figura 3.12.

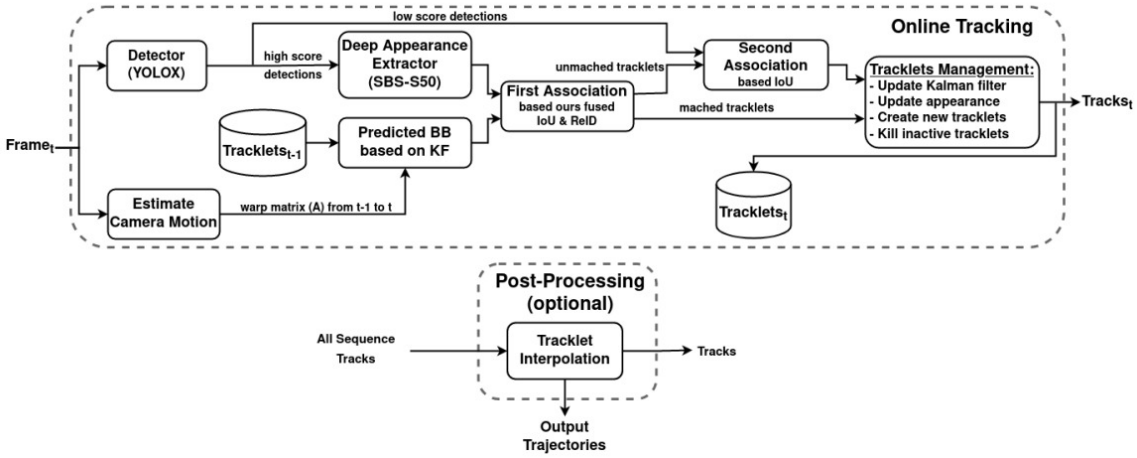


Figura 3.12: Arquitetura do BoT-SORT. Fonte: [Aharon, Orfaig e Bobrovsky 2022]

O BoT-SORT propõe estratégias para lidar com problemas comuns em trackers como o SORT, e as integra no ByteTrack [Zhang et al. 2022]. As inovações do BoT-SORT são:

- **Filtro de Kalman Modificado**: Para modelar o movimento dos objetos, o vetor de estado do filtro de Kalman é ajustado. Em vez de prever apenas a relação de aspecto da caixa delimitadora, o filtro agora estima diretamente a largura e a altura, melhorando a precisão do rastreamento. Ele também adapta as matrizes de covariância de ruído para serem dependentes do tempo, tornando o filtro mais robusto em condições variáveis.
- **Compensação de Movimento de Câmera (CMC)**: Para melhorar o desempenho em cenários com movimento de câmera, o BoT-SORT introduz uma técnica de compensação de movimento baseada na transformação afim para corrigir a posição da caixa delimitadora de cada objeto entre os quadros. Essa compensação ajusta o movimento da câmera, permitindo que o rastreador mantenha a precisão ao associar detecções, mesmo com deslocamentos grandes ou mudanças rápidas de perspectiva.
- **Fusão IoU-ReID**: Para aumentar a robustez do rastreamento, o método combina informações de similaridade de aparência e de posição. Ele emprega uma combinação adaptativa entre a distância IoU (informação de movimento) e a distância cosseno (similaridade visual) para rejeitar associações improváveis e selecionar as mais confiáveis. Isso reduz falhas de rastreamento e aumenta a precisão em cenários de alta densidade, onde a aparência e a posição podem variar.

CAPÍTULO 4

Metodologia

Este trabalho propõe combinar informações visuais e de fluxo óptico para uma seleção coerente de regiões de imagem onde a odometria visual pode ser aplicada de forma menos sensível à presença de objetos dinâmicos. O texto a seguir apresenta o *dataset* escolhido para os experimentos deste trabalho, bem como a justificativa de escolha, preparação e ajustes de anotação. Em seguida, o texto mostra uma arquitetura de rede neural artificial para a segmentação de objetos dinâmicos e como a informação de fluxo óptico é integrada ao fluxo de processamento da rede.

4.1 Escolha do Dataset

A escolha do dataset foi orientada pela disponibilidade de imagens em ambientes urbanos que incluam anotações de objetos dinâmicos e ground truth para avaliação de odometria. Idealmente, o dataset deve ter alta variação de cenários, e grande quantidade e variação de objetos dinâmicos observados.

O artigo SMSNet [Vertens, Valada e Burgard 2017] introduz dois datasets para este propósito: O KITTI-Motion e Cityscapes-Motion, e o Cityscapes-Motion não pode ser acessado devido a sua página de download estar fora do ar. Outro dataset para esse propósito é o WHUVID [Chen et al. 2022].

O KITTI-Motion contém anotação de classe semântica por pixels e de objetos em movimento para 255 imagens do *dataset* KITTI Raw. As imagens têm resolução de 1280x384 pixels e contêm cenas de rodovias, áreas residenciais e centros urbanos. A Figura 4.1 mostra um exemplo de imagem do *dataset*. Sua vantagem é que o dataset já está preparado para treinar uma rede neural de segmentação de objetos dinâmicos, sem necessidade de anotações adicionais. Sua maior limitação é o baixo número de imagens anotadas (apenas 255) e baixa variação de veiculos e ambientes.

Figura 4.1: KITTI-Motion

O *dataset* WHUVID [Chen et al. 2022] tem como objetivo ser um *dataset* mais adequado para avaliar algoritmos de odometria visual-inercial, ao alcançar uma quantidade de imagens maior que *datasets* anteriores, com maior variabilidade de ambientes e de veículos presentes, com alta taxa de aquisição de imagens (30Hz), e incluindo cenários que raramente são encontrados em outros datasets, como alta densidade de arranha-céus, ruas largas com muitas pessoas, túneis e pontes. Ele inclui dados de câmera externa, IMU e *ground truth* a partir de GPS. O *dataset* inclui 336000 imagens divididas em 34 sequências capturadas na cidade de Wuhan, China. Ele inclui 682000 anotações de detecção de objetos para as classes carro, pessoa, bicicleta (incluindo motos) e sinais de tráfego. O gráfico na figura 4.2 mostra a frequência desses objetos por sequencia.

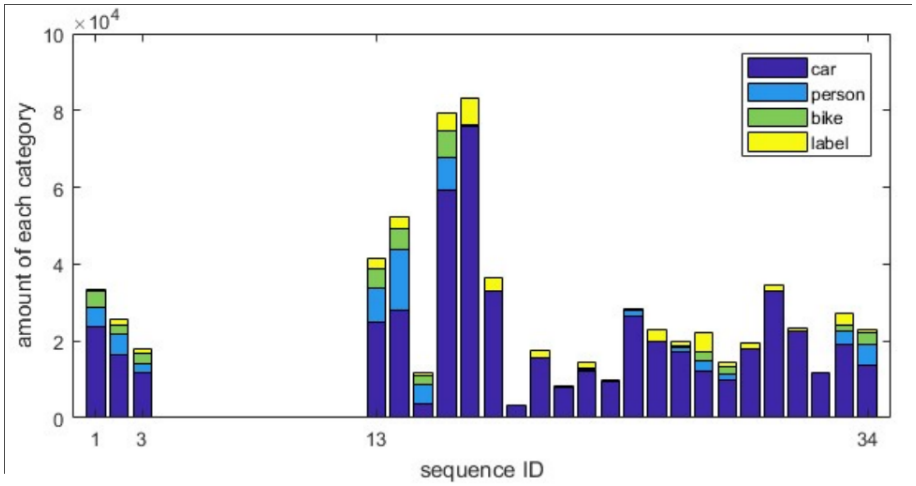


Figura 4.2: Frequência de classes anotadas por sequência no WHUVID

O conjunto de figuras 4.3 mostra alguns exemplos de imagens e anotações do WHUVID.

Figura 4.3: Imagens com anotações das sequências 01, 02, 17 e 19 do WHUVID

Uma limitação do uso desse *dataset* neste trabalho é a ausência de segmentação semântica dos objetos e a distinção entre objetos estáticos e dinâmicos. Por esse motivo, para atender os objetivos deste trabalho, foi necessário adaptar o *dataset* com a inclusão de máscaras de segmentação semântica e classificação de objetos estáticos e dinâmicos. A seção 4.2 detalha como essas informações foram agregadas ao *dataset*.

Devido ao alto número de imagens e variação de cenários e grande quantidade de objetos dinâmicos, o WHUVID foi escolhido como o *dataset* principal deste trabalho. No entanto, para aumentar a diversidade de dados, o *dataset* KITTI-Motion também foi adicionado.

4.2 Preparação de Dataset

4.2.1 Script de anotação de objetos dinâmicos

O *dataset* WHUVID não inclui a distinção entre objetos estáticos e dinâmicos. A anotação de carros inclui carros em movimento, mas também inclui carros parados e estacionados na mesma categoria. Caso seja feito treinamento com esses dados, a rede neural pode ser capaz de segmentar carros mas não de detectar se o carro está em movimento ou não. Pra isso é necessário uma anotação adicional marcando o carro como parado ou em movimento.

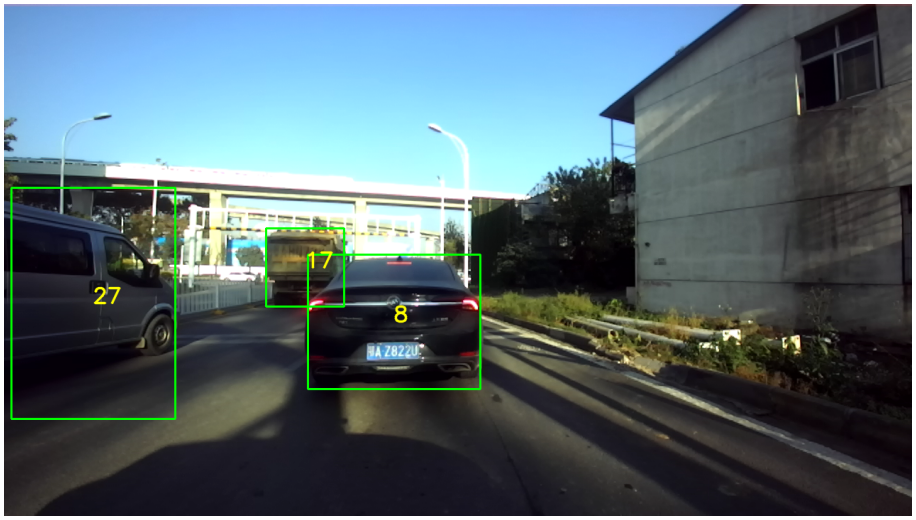
A anotação manual de cada imagem desse *dataset* exige esforço e muito tempo de trabalho devido ao grande número de imagens. Por esse motivo, optou-se por criar uma metodologia de anotação semi-automática para o enquadramento dos dados na proposta

final deste trabalho. Como o *dataset* é formado por sequências de imagens enquanto um veículo percorre seu trajeto, a partir da suposição de que os objetos na cena mantém seu movimento constante em um curto período de tempo, a informação de um quadro pode ser propagada para os quadros seguintes com o auxílio de algoritmos de *tracking*.

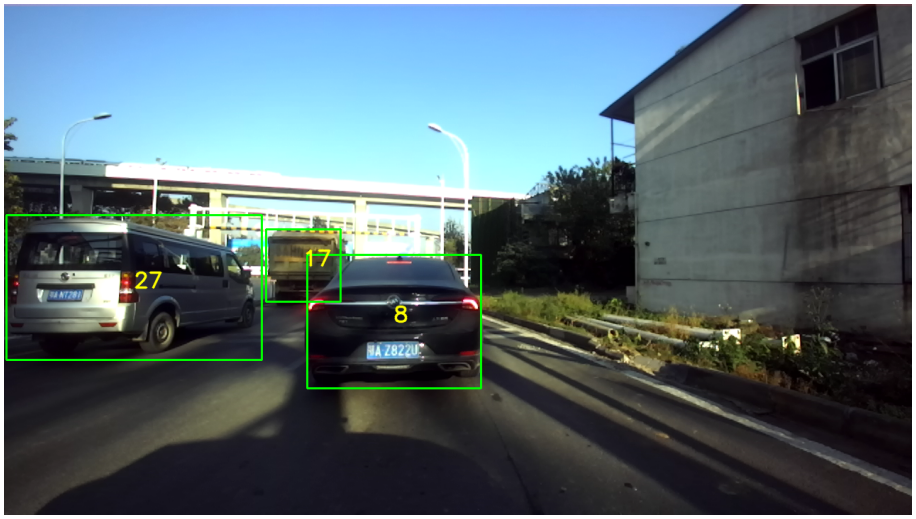
Desse modo, objetos em movimento tendem a se manter em movimento e objetos parados tendem a continuar parados, esses rótulos precisam ser definidos uma única vez. Para outros, o rótulo precisa ser alterado apenas quando o objeto apresenta uma transição do seu movimento. Aproveitando essa regularidade, foi desenvolvido um *script* em Python que permite selecionar automaticamente um mesmo objeto em diferentes quadros com o uso da informação de *tracking*. Uma vez selecionado, é possível classificar toda a sequência do objeto como estático ou dinâmico.

A primeira etapa do processo de adaptação dos dados é aplicar um algoritmo de *tracking* dos objetos da cena para identificar as *bounding boxes* que pertencem ao mesmo objeto ao longo do vídeo. Ese problema é descrito na seção 3.5. Para isso, foram verificados os algoritmos SORT [Bewley et al. 2016], Deep-Sort [Wojke, Bewley e Paulus 2017], StrongSORT [Du et al. 2023] e BoT-SORT [Aharon, Orfaig e Bobrovsky 2022], sendo que o BoT-SORT, além de ser um algoritmo recente, é competitivo com o estado da arte de tracking em ambientes urbanos complexos, apresentando melhores resultados nos experimentos com o *dataset* WHUVID. O BoT-SORT é descrito em mais detalhes na seção 3.5. A entrada do BoT-SORT são as imagens e as *bounding boxes* já presentes na anotação original do WHUVID, das classes pessoa, carro e bicicleta.

A Figura 4.4a mostra uma imagem da sequencia 25 do WHUVID com numeros indicando a qual objeto do BoT-SORT a *bounding box* pertence. Avançando cinco imagens na sequência, na Figura 4.4b, os objetos mantém o mesmo id.



(a) Imagem com *bounding box* e IDs do BoT-SORT



(b) Imagem com *bounding box* e IDs do BoT-SORT, cinco frames após a figura 4.4a

Figura 4.4: Exemplo de tracking do algoritmo BoT-SORT. Na imagem esquerda a detecção dos objetos com a apresentação da boudingbox e seus respectivos labels. Na imagem da direita, a detecção dos mesmos objetos da mesma cena 5 frames à diante.

Ao completar o *tracking*, a interface de usuário mostra a primeira imagem da sequência com as *bounding boxes* sobrepostas, representando objetos estáticos pela cor verde e objetos em movimento pela cor azul, como na Figura 4.5.

Inicialmente, todos os objetos são considerados estáticos. O usuário então clica na *bounding box* para marcá-la como objeto dinâmico. Caso a *bounding box* seja marcada, o objeto será considerado como dinâmico em todos as imagens a seguir até que o usuário marque alguma imagem seguinte como estático. O usuário pode retornar ou avançar frame a frame, ou pode pressionar play para reproduzir o vídeo em tempo real e pausar quando quiser marcar o próximo objeto. Dessa forma, a maioria das imagens não precisa de anotação individual, acelerando o processo de anotação.

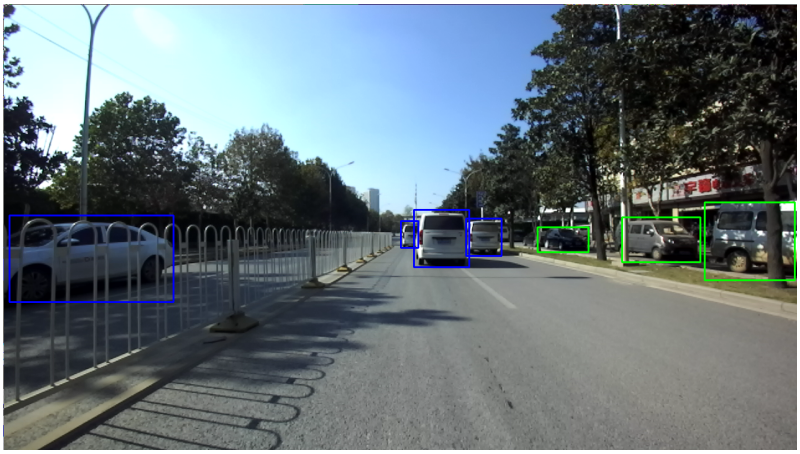


Figura 4.5: Carros anotados como estático ou dinâmicos no WHUVID

Um exemplo pode ser visto na Figura 4.6, em que o objeto de id 27 em movimento foi marcado como dinâmico. Ao avançar cinco imagens, na Figura 4.7, a *bounding box* do objeto de id 27 já esta marcado como dinâmico e os outros permanecem como estáticos, sem necessidade de alteração pelo anotador. Quando o carro parar, basta o anotador marcá-lo como estático a partir da imagem em que ele para, e o objeto será considerado como estático nas imagens seguintes. Igualmente se o objeto estava parado e começou a se movimentar.

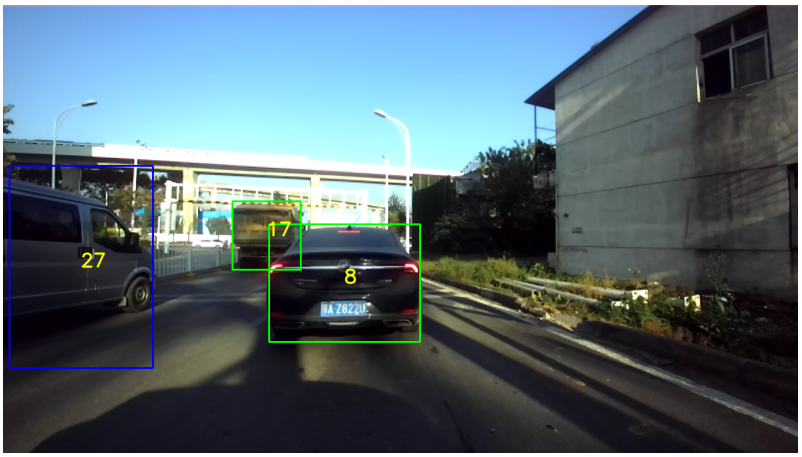


Figura 4.6: Imagem com ids de objeto em amarelo

Esse script foi usada para anotar as sequências do WHUVID utilizadas neste trabalho, que são as sequências 01, 02, 17, 19, 20, 23, 24, 25, 30, 31 e 32 para treino, e as sequências 03, 18 e 22 para avaliação. A maioria das sequências foram escolhidas por conter alto número de carros em movimento, e algumas por ter alto número de carros estacionados.

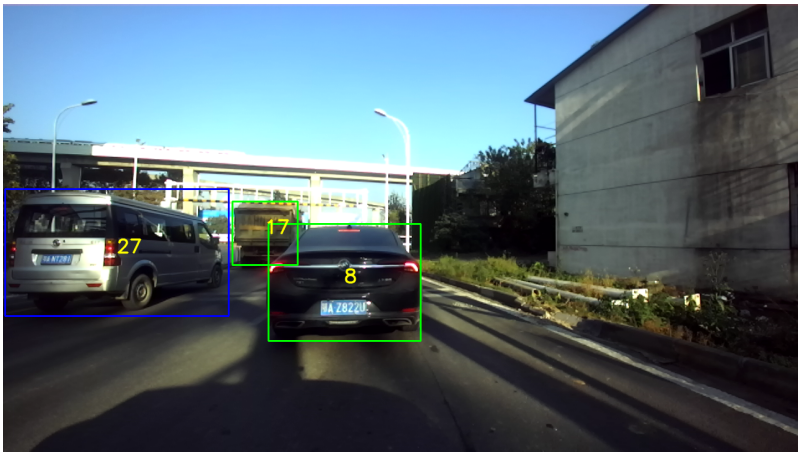


Figura 4.7: Imagem com ids de objeto em amarelo

4.2.2 Geração de máscaras de segmentação

Outra limitação do WHUVID para o contexto deste trabalho é que o *dataset* não inclui a mascara de segmentação de cada objeto, fornecendo apenas *bounding boxes* dos objetos possivelmente dinâmicos, como na Figura 4.8.

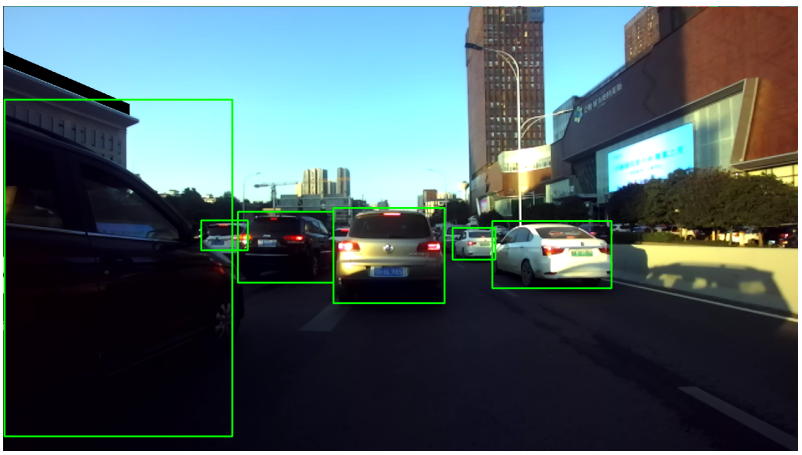


Figura 4.8: Anotação do WHUVID

Recentes avanços em redes de segmentação possibilitaram obter máscaras pixel a pixel a partir da *bounding box*. O Segment Anything Model (SAM) [Kirillov et al. 2023] apresenta excelente capacidade de generalização em cenários diversos e acurácia em bordas de segmentação. Ele é um modelo treinado de forma semi-supervisionada de forma a permitir aplicações em *datasets* que não pertencem ao seu conjunto de treinamento. O Segment Anything é explicado em mais detalhes na seção 3.3.1.

O modelo Segment Anything foi aplicado ao *dataset* WHUVID para gerar as mascaras de segmentação a partir da *bounding box* e mostrou bons resultados. A Figura 4.9 mostra uma imagem de exemplo com anotação de *bounding box*, e a máscara de segmentação obtida com o SAM.

Figura 4.9: *Bounding box* e mascaras obtidas com o Segment Anything

Foi feito inferência para as 98491 imagens das 14 sequências do WHUVID anotadas usando o script de anotação descrito na seção 4.2.1. Essas máscaras foram usadas como *ground truth* para treinar a rede neural de segmentação.

4.3 Fluxo Óptico

O fluxo óptico, discutido em mais detalhes na seção 3.4, fornece informações sobre o movimento relativo entre os objetos e a câmera. Tal informação é necessária para classificá-los como estáticos ou dinâmicos. Artigos como [Maczyta, Bouthemy e Meur 2019] indicam que as regiões da imagem correspondentes a objetos em movimento apresentam fluxo óptico diferente das regiões ao seu redor, o que pode ajudar a rede neural a diferenciar objetos estáticos de dinâmicos.

Foram feitos experimentos com o fluxo óptico de [Farnebäck 2003] implementado no OpenCV. Seus parâmetros foram ajustados até que o movimento dos carros sejam perceptíveis na imagem resultante, mas o método ainda apresentou limitações. Carros distantes não apresentam fluxo óptico bem definido, e há muito ruído. Já os métodos que utilizam redes neurais apresentaram bons resultados, até com carros distantes e ambientes mais complexos. Após testar vários modelos, foi escolhido o DICL [Wang et al. 2020] por ser treinado no KITTI e ter apresentado resultado bom no WHUVID. O DICL é apresentado na seção 3.4 deste trabalho.

A Figura 4.10 mostra um exemplo de fluxo óptico na sequencia 03 do WHUVID, em que a cor indica a direção do movimento e a intensidade indica o tamanho do deslocamento de cada píxel.

Figura 4.10: Fluxo Óptico

A imagem gerada pelo DICL é usada como uma das entradas da rede neural de segmentação, junto com a imagem RGB.

4.3.1 Rede Neural de segmentação de objetos dinâmicos

A rede neural combina informações visuais e de deslocamento para uma seleção coerente de regiões de imagem onde a odometria visual pode ser aplicada de forma menos sensível à presença de objetos dinâmicos.

A saída é uma máscara de segmentação para três classes:

1. Objeto em movimento
2. Objeto estático
3. Outros (plano de fundo, objetos não anotados)

Foi necessário adicionar a classe "Objeto estático"para que a rede neural aprenda a diferenciar entre objetos estáticos e dinâmicos. Caso "Objeto em movimento"seja a única classe e os objetos parados sejam tratados como "Outros", ela aprende apenas a segmentar as classes presentes no *dataset* (carros, motos, pessoas), independente de estarem em movimento ou não.

A arquitetura pode ser visualizada na imagem 4.11 .

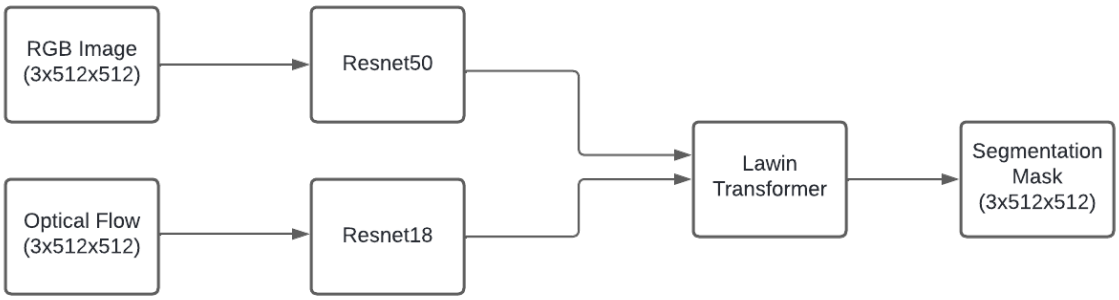


Figura 4.11: Arquitetura da rede neural

A resnet50 é utilizada para extrair as *features* da Imagem RGB, aproveitando de seu pré-treinamento para que não seja necessário uma grande quantidade de imagens para obter um bom desempenho no componente visual da rede. Já as *features* do fluxo óptico são extraídas pela Resnet18. Não é necessário uma rede muito grande, porque os padrões presentes no fluxo óptico não são muito complexos. O *decoder* é o Lawin transformer [Yan, Zhang e Wu 2022], escolhido por ser um método recente que apresentou bons resultados em segmentação. Foi feita uma pequena modificação no encoder. Ele já aceita múltiplas entradas, na forma de múltiplos estágios do encoder da imagem RGB. Foi necessário apenas fornecer os estágios da Resnet18 em conjunto com os estágios da Resnet50 para que o Lawin considere as duas entradas na segmentação.

A função de perda utilizada foi a *focal loss* [Lin et al. 2020], definida na equação 4-1, por ser projetada para lidar melhor com desequilibrio de classes. Tal desequilíbrio é presente no WHUVID, porque há muito mais píxeis de plano de fundo do que de classes anotadas, e dentro dessas classes, mais objetos em movimento do que estáticos.

$$FL(p_t) = -\alpha \cdot (1 - p_t)^{\gamma} \cdot log(p_t) \tag{4-1}$$

com $\alpha = 0.75$ e $\gamma = 2$. Nesta equação, p_t é a probabilidade estimada pelo modelo para a classe correta. α é o peso que equilibra a importância de casos positivos vs negativos, e γ define o quanto o modelo foca em problemas mais difíceis.

O modelo foi então treinado em 77930 imagens referentes as sequências 01, 02, 17, 19, 20, 23, 24, 25, 30, 31 e 32 do WHUVID, e nas 255 imagens do Kitti-Motion. Já as sequências 03, 18 e 22 do WHUVID, com o total de 20561 imagens, foram separadas para avaliar o modelo em ambientes não vistos anteriormente. Os resultados do treinamento são apresentados no capítulo 5.

CAPÍTULO 5

Resultados

5.1 Segmentação de objetos dinâmicos

A rede neural desenvolvida foi avaliada nas sequências 03, 18 e 22 do WHUVID por serem capturados em locais diferentes entre si. A sequencia 03 tem muitos carros parados, mas poucos em movimento. A sequencia 18 tem uma grande quantidade de carros em movimento, enquanto a sequencia 22 mistura carros parados e em movimento. O total de imagens é de 20561.

A tabela 5.1 mostra os IoUs de cada classe em treinamento e avaliação:

Tabela 5.1: IoU de treino e avaliação

	Treino	Avaliação
IoU de objetos dinâmicos	0.886	0.828
IoU de objetos estáticos	0.769	0.576
IoU de outros (plano de fundo, etc)	0.993	0.994

Os resultados indicam que, no *dataset* de teste, a rede neural consegue segmentar as classes anotadas, como carros e pedestres. O IoU para objetos dinâmicos é bom, mas os objetos estáticos apresentam uma taxa de erro maior. Dentre eles, o mais importante é segmentar os objetos em movimento, pois são eles que são usados para filtrar a odometria visual.

A Figura 5.1 mostra a predição e anotação em uma imagem da sequencia 22 do WHUVID. Nesta visualização, a imagem RGB foi combinada com a máscara de segmentação. Objetos azuis são considerados dinâmicos e objetos verdes são estáticos.

Figura 5.1: Resultado na sequência 22.

Na Figura 5.2 pode ser visto outro resultado, na sequência 03 do WHUVID. Há alguns erros, mas a maioria dos objetos dinâmicos são identificados.

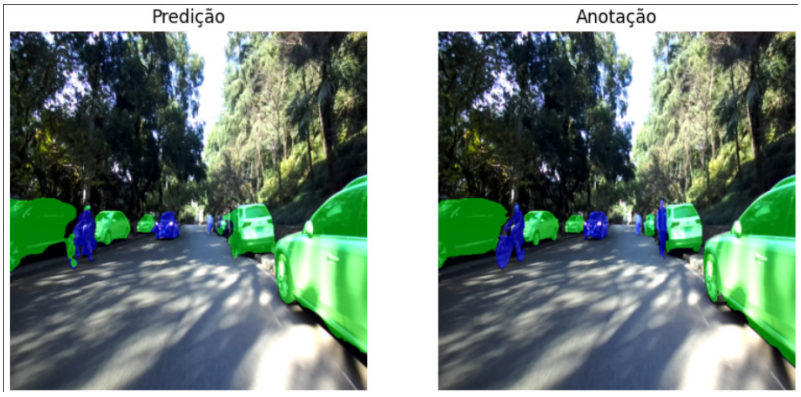


Figura 5.2: Resultado na sequência 03

A Figura 5.3 mostra um resultado na sequência 18. As imagens foram capturadas em um ambiente mais simples, em que todos os objetos na avenida são carros estão em movimento. Por isso, o desempenho neste ambiente é melhor que nos outros.

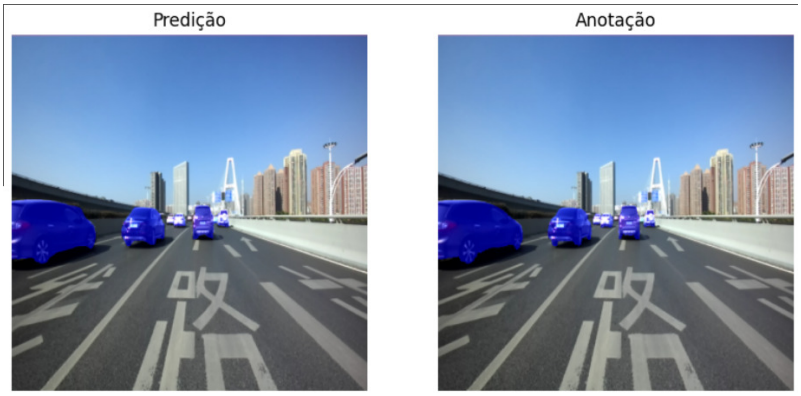


Figura 5.3: Resultado na sequência 18

Na Figura 5.4, um carro está em movimento em baixa velocidade, e é segmentado corretamente como objeto em movimento.

Figura 5.4: Inferência com carro em movimento

Poucos segundos depois, na Figura 5.5, este mesmo carro para na faixa de pedestre, e a rede de segmentação muda a classificação de todos os seus pixeis para objeto estático.

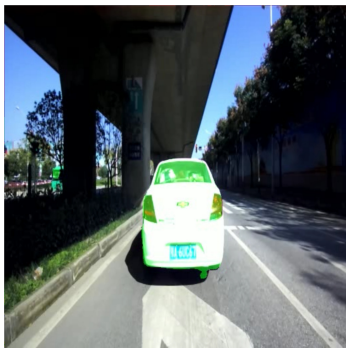


Figura 5.5: Inferência com carro parado

Estes resultados indicam que a rede neural consegue utilizar os dados de fluxo óptico para distinguir objetos estáticos de dinâmicos, quando há poucas informações visuais em uma única imagem RGB para tal classificação.

Também foi medido o tempo de execução dos modelos, executado em uma placa de vídeo RTX 4080, na tabela 5.2

Tabela 5.2: Tempo de execução

Componente	Tempo
Tracking (Bot-SORT)	30ms
Segment Anything	95ms
Fluxo óptico	75ms
Modelo de Segmentação	14ms

O Bot-SORT e o Segment Anything são utilizados apenas na fase de anotação, e por isso seu tempo de execução não afeta o tempo de inferência total do sistema. Para 100 mil imagens, o *tracking* demora menos de 1 hora, e a etapa do Segment Anything, que é executado duas vezes por imagem (uma vez para gerar máscara de objetos estáticos e uma para objetos dinâmicos), leva 5.3 horas ao total.

Apesar de a execução em tempo real não ser um dos objetivos deste trabalho, o modelo de segmentação apresentou um bom tempo de inferência. Já a rede de fluxo óptico leva muito mais tempo. O total de 89ms de inferência (fluxo óptico + segmentação) permite sua aplicação em uma odometria com taxa de atualização de aproximadamente 11Hz. Em um trabalhos futuros, pode ser feita a exploração de métodos de fluxo óptico mais leves para tornar a rede mais adequada para aplicações em tempo real.

5.2 Experimentos com odometria

Para avaliar o efeito do filtragem de pontos dinâmicos em odometria, o filtro foi integrado a um sistema de odometria visual. Idealmente, o sistema de odometria visual deve ser monocular, capaz de estimar a trajetória com escala relativa, e com implementação de código aberto. O algoritmo open-source pyslam, descrito na seção 3.2.2, atende esses requisitos, e foi escolhido para esses experimentos. O algoritmo foi testado no *dataset* WHUVID, que fornece a calibração de câmera e ground truth. O ORB-SLAM2 monocular [Mur-Artal e Tardós 2017] também foi testado e comparado com o pyslam. Ambos algoritmos apresentaram dificuldades de estimar bem o movimento no WHUVID, falhando especiamente em manter a escala consistente. Apesar disso, o impacto da rede neural proposta pode ser avaliada comparando a acurácia do pylsma com o filtro, com pyslam original e o ORB-SLAM2.

Inicialmente, é executada a inferência para todas as imagens da sequência. Para cada imagem no *dataset*, é gerada uma máscara de segmentação, salva em imagens .jpg em escala de cinza, em que um píxel de valor 0 representa o plano de fundo, o valor 255 representa objetos em movimento e o valor 128 representa objetos segmentados que estão parados no momento.

O algoritmo pyslam foi modificado para ler essas máscaras, e após a etapa de identificação de pontos característicos, os pontos de valor 255 na máscara, classificados como em movimento, são removidos. O resto do algoritmo procede sem alteração, realizando a odometria apenas nos pontos estáticos.

O pyslam foi executado para as sequências 18 e 22 do WHUVID, com e sem a aplicação do filtro. A odometria do ORB-SLAM também foi executado para as mesmas sequências. A tabela 5.3 mostra o erro de pose médio nas duas sequências.

Tabela 5.3: Erro médio de pose do pyslam, pyslam com filtro e ORB-SLAM2

	Sequencia 18	Sequencia 22
pyslam	414.1m	230.7m
pyslam + filtro	283.7m	135.1m
ORB-SLAM2	1734m	149.4m

Estas sequências foram escolhidas por apresentarem um grande quantidade de objetos em movimento. Nos testes, o pyslam com filtro apresentou erro menor que o pyslam original e que o ORB-SLAM2 nestas sequências. Vale destacar que o ORB-SLAM2 não apresentou fechamento de loop nas sequências testadas, e por isso ele pode apresentar melhor desempenho que o psyam com filtro em ambientes diferentes.

A sequência 22 do WHUVID ocorre em um ambiente urbano, com grande quantidade de carros em movimento e estacionados. Nenhuma imagem desta sequência está no *dataset* de treinamento. O trajeto estimado pelos três algoritmos é apresentado na Figura 5.6, enquanto a orientação é mostrada na Figura 5.7.

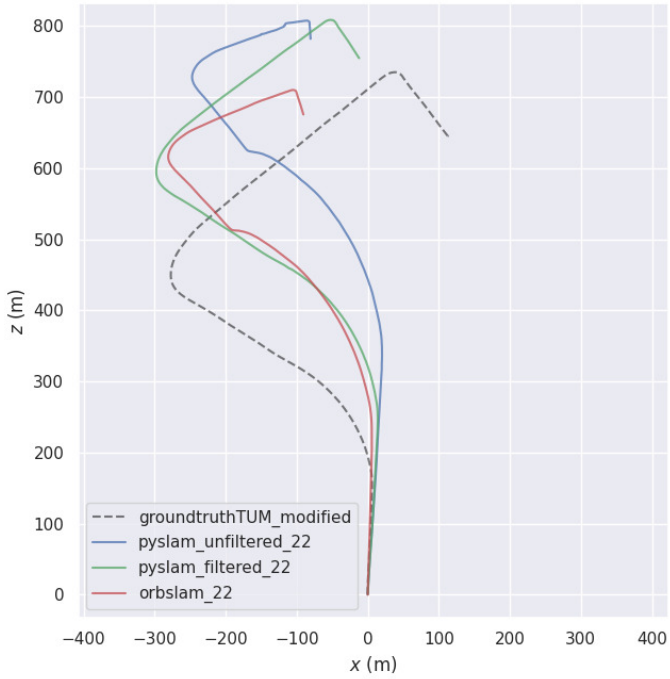


Figura 5.6: Trajeto estimado do pyslam com o filtro (verde), pyslam sem o filtro (azul) e ORB-SLAM2 (vermelho)

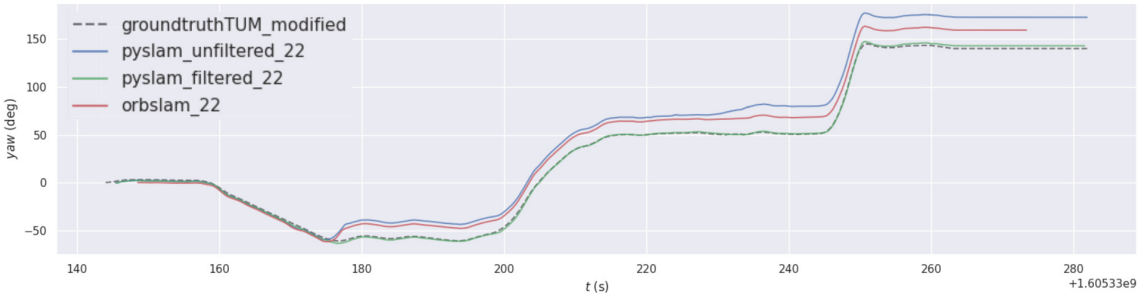


Figura 5.7: Estimativa de orientação do pyslam com o filtro (verde), pyslam sem o filtro (azul) e ORB-SLAM2 (vermelho)

O filtro melhora significantemente a trajetória estimada do pyslam, especialmente quando há um carro em movimento próximo a camera. Próximo do timestamp de 180 +1.6e9 segundos na Figura 5.7, há um desvio grande de orientação do pyslam e do ORB-SLAM2, enquanto o pyslam com filtro se mantém próximo do ground truth. Este é um momento em que um veículo se aproxima muito da câmera, como vistos nas Figuras 5.8 e 5.9, da imagem e pontos rastreados neste momento.

Figura 5.8: Pontos rastreados pelo pyslam sem a aplicação do filtro

Pode ser visto na figura 5.8 que o pyslam calcula a odometria utilizando um grande número de pontos na van em movimento. Esses pontos não condizem com o movimento real do veículo, o que causa os erros de estimativa em 180 +1.6e9 segundos na Figura 5.7,

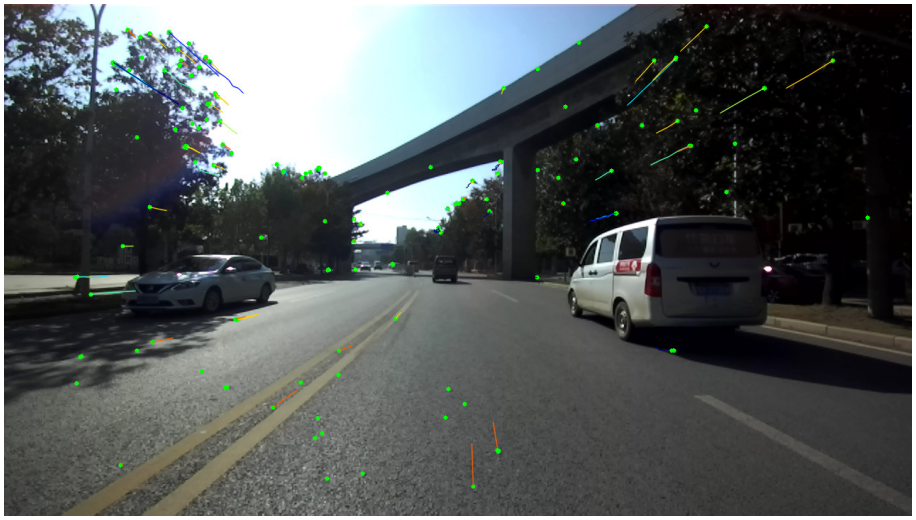


Figura 5.9: Pontos rastreados pelo pyslam com a aplicação do filtro

Já na figura 5.9, o filtro foi capaz de remover os pontos característicos da van em movimento, e como a odometria é calculada apenas com os pontos estáticos, a estimativa se aproxima mais do ground truth. Uma limitação apresentada pelo filtro é que ele não é capaz de remover o ponto característico presentes na sombra da van, mesmo estando em movimento, por estar fora do objeto segmentado. O número de pontos característicos na sombra é pequeno, e por isso não houve erros de estimativa devido a sombra.

A figura 5.10 mostra os pontos característicos calculados pelo ORB-SLAM2.

Figura 5.10: Pontos rastreados pelo ORB-SLAM2

Assim como o pyslam, o ORB-SLAM2 não foi capaz de remover os pontos característicos da van e no carro, e por isso também apresentou os mesmos erros de estimativa nesta região da sequência. O segundo grande erro de estimativa do pyslam sem o filtro ocorre no timestamp de 230+1.6e9 segundos na Figura 5.7. Os pontos rastreados pelo pyslam podem ser vistos nas imagens 5.11.

Figura 5.11: Pontos rastreados pelo pyslam sem a aplicação do filtro

Novamente o pyslam não foi capaz de remover os pontos característicos da van em movimento, e por isso, obteve uma estimativa de orientação muito diferente do ground truth. O pyslam com filtro removeu corretamente estes pontos, como visto na figura 5.12, e por isso apresentou uma estimativa de orientação melhor.

Figura 5.12: Pontos rastreados pelo pyslam com a aplicação do filtro

Os algoritmos também foram testados na sequencia 18 do WHUVID, mostrado na Figura 5.13 e orientação na Figura 5.14

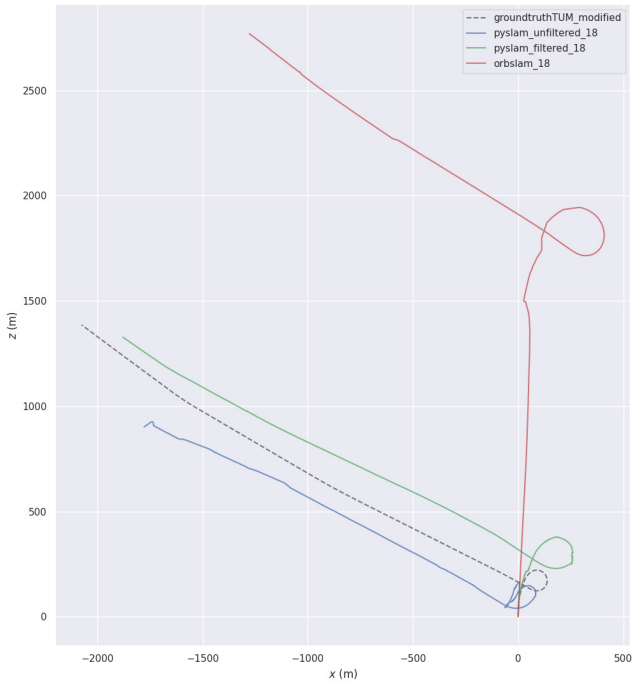


Figura 5.13: Trajeto estimado do pyslam com o filtro (verde), pyslam sem o filtro (azul) e ORB-SLAM2 (vermelho)

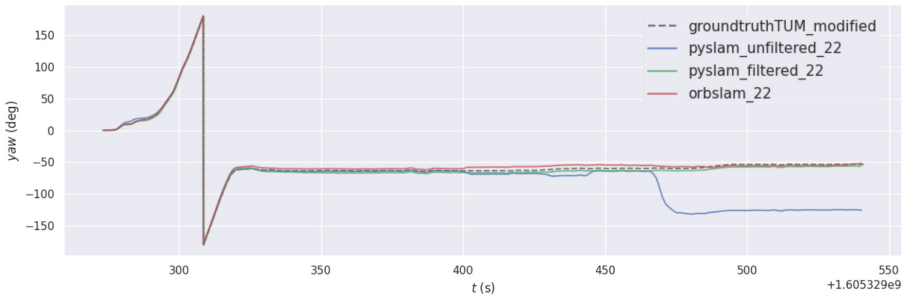


Figura 5.14: Estimativa de orientação do pyslam com o filtro (verde), pyslam sem o filtro (azul) e ORB-SLAM2 (vermelho)

A sequencia 18 ocorre em uma avenida com muitos carros em movimento, sendo mais difícil de estimar que a sequência 22. Há uma grande diferença na estimativa de odometria do pyslam com e sem o filtro, e ambos se aproximam mais do ground truth que o ORB-SLAM2. A figura 5.15 mostra os pontos rastreados pelo pyslam nessa sequência.

Figura 5.15: Pontos rastreados pelo pyslam, sem a aplicação do filtro

O pyslam sem o filtro novamente não é capaz de remover os pontos característicos dos carros em movimento, e seleciona poucos pontos na rua e no viaduto. Por isso, boa parte dos pontos utilizados não são adequados para estimar o movimento do veículo, e a acurácia da odometria é afetada.

Os pontos característicos utilizados pelo pyslam com da aplicação do filtro é visto na figura 5.16.

Figura 5.16: Pontos rastreados pelo pyslam, com a aplicação do filtro

Com a aplicação do filtro, além de remover os pontos característicos dos carros em movimento, o pyslam também foi capaz de selecionar pontos melhores para estimar a odometria. Foram escolhidos muitos pontos na rua e no viatudo, e por isso, os pontos característicos utilizados representam melhor o movimento do carro.

Os pontos rastreados pelo ORB-SLAM2 podem ser vistos na Figura 5.17.

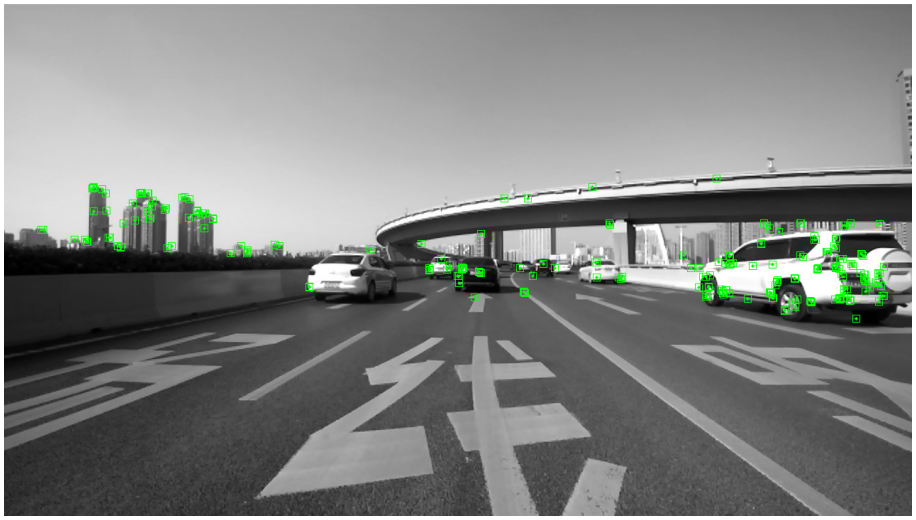


Figura 5.17: Pontos rastreados pelo ORB-SLAM2

O ORB-SLAM2 estimou a odometria utilizando uma grande quantidade de pontos nos carros em movimento e ainda menos pontos na rua e viaduto, e por isso a sua odometria foi a pior dos três. Estas imagens ocorrem no início da sequência, dificultando a estimativa das odometrias sem o filtro de pontos dinâmicos. A orientação das três odometrias se mantiveram próximas (Figura 5.14), com exceção do pyslam sem filtro, que estima incorretamente perto do final do trajeto.

Os testes demonstraram que o filtro de pontos dinâmicos é capaz de remover corretamente estes pontos indesejados da odometria do pyslam, e também contribui para uma melhor seleção de pontos característicos para calculo de odometria. A estimativa do pyslam apresentou grandes melhoras em ambientes com grande quantidade de objetos dinâmicos no dataset do WHUVID, e obteve uma estimativa melhor que o ORB-SLAM2 nas sequências testadas.

CAPÍTULO 6

Conclusão

Este trabalho propôs a filtragem de regiões dinâmicas para aprimorar a qualidade dos pontos usados em algoritmos de Odometria Visual. Diante dessa proposta, este trabalho buscou combinar informações de imagens RGB e fluxo óptico como fonte de informações para o treinamento de um modelo neural capaz de distinguir regiões estáticas e dinâmicas em um contexto onde a própria câmera está em movimento. Para atingir esse objetivo, foi necessário agregar informações a *datasets* já disponíveis, sendo também proposta uma metodologia de anotação semi-automática de regiões dinâmicas.

A metodologia de anotação desenvolvida neste estudo possibilita uma adaptação mais eficiente de um conjunto de dados de detecção de objetos para a tarefa de segmentação de movimento. Caso a ferramenta seja adotada pela comunidade científica, ela pode contribuir para o aumento do número de *datasets* disponíveis para aplicações afins a este trabalho.

As máscaras de segmentação de movimento foram adicionadas a um grande número de sequências do *dataset* WHUVID, resultando em um grande volume de dados para a segmentação de movimento.

A arquitetura de rede neural de segmentação de objetos dinâmicos proposta se mostrou capaz de realizar a segmentação de objetos dinâmicos, a partir da combinação de imagens e fluxo óptico, mesmo em situações onde há movimentos diferentes em relação à câmera. Os experimentos mostraram bons resultados em segmentar imagens do *dataset* WHUVID.

Os experimentos demonstraram que a metodologia desenvolvida neste trabalho é capaz de aprimorar a estimativa de algoritmos de odometria visual mesmo em metodologias menos robustas, uma vez que pontos dinâmicos interferem consideravelmente na estimativa do movimento. Os resultados obtidos foram capazes de superar algoritmos bem estabelecido como o ORB-SLAM2 no *dataset* escolhido como referência desta pesquisa.

O código da ferramenta de anotação do *dataset*, de geração de máscaras por Segment Anything, de inferência de fluxo óptico pelo DICL, de treinamento e de inferência da rede de segmentação e os pesos da rede de segmentação estão disponíveis em https://github.com/thiagohenrique1/dynamic_object_segmentation.

6.1 Trabalhos futuros

Com a introdução da ferramenta de anotação rápida, é mais fácil preparar *datasets* novos para aplicações em ambientes dinâmicos. Isso é importante devido ao baixo número de *datasets* disponíveis na literatura para ambientes dinâmicos. Ele facilita que trabalhos futuros introduzam *datasets* em locais diferentes. Cada país tem características diferentes que podem afetar o desempenho de algoritmos de odometria visual, como a presença de buracos na estrada, maior variação de veículos, frequência diferente de árvores e de prédios. Por isso, uma possibilidade de trabalho futuro é o desenvolvimento de um *dataset* brasileiro para odometria visual em ambientes dinâmicos. Ao treinar uma rede neural em um contexto nacional, ela pode entender melhor as particularidades das ruas e avenidas brasileiras, e identificar melhor os objetos mais comuns neste país.

A introdução de sensores adicionais também pode contribuir para melhor distinção entre objetos estáticos e dinâmicos. Dados de IMU podem auxiliar na interpretação do fluxo óptico, por fornecer uma referência de movimento do veículo independente da câmera. Como o IMU não é afetado pela presença de objetos em movimento no ambiente, ele pode auxiliar em distinguir qual movimento observado é devido a câmera e qual é devido a objetos dinâmicos. Outra modificação possível seria o uso de câmera estéreo ou câmera de profundidade, para obter também a escala absoluta do ambiente e a distância real destes objetos a câmera.

Podem ser realizados experimentos com o posicionamento da câmera, como câmeras laterais, traseiras ou na diagonal, para avaliar se o desempenho da rede neural se mantém adequado nestas situações. O treinamento com câmeras em posições diferentes pode também levar a uma melhor generalização da rede neural, para que ela entenda melhor o movimento da câmera e dos objetos. A redundância de câmeras em perspectivas diferentes na mesma rede neural também pode levar uma compreensão melhor do ambiente.

A rede de segmentação desenvolvida usa informações de apenas duas imagens em sequência, sendo a segunda usada apenas para gerar o fluxo óptico. É possível fazer experimentos com o uso de *encoders* de vídeo, como o Swin-Video-Transformer [Liu et al. 2022], que podem se aproveitam de uma janela maior de imagens para entender melhor o movimento da câmera e dos veículos.

A aplicação em sistemas de tempo real não foi um dos objetivos deste trabalho. Todos os experimentos com odometria foram feitos de forma offline. Apesar disso, as medidas de tempo de execução mostraram que, com 89 milissegundos de tempo de inferência, a rede neural pode ser executada de forma iterativa com uma frequência de aproximadamente 11Hz em um *desktop* de grande poder computacional. Para que ela se torne mais adequada para aplicações em tempo real, seria ideal fazer modificações

para reduzir o custo computacional para a execução em máquinas com placas de vídeo mais fáceis de embarcar em veículos autônomos. Uma área de investigação é o uso de algoritmos de fluxo óptico mais leves, já que o DICL foi o componente com maior tempo de execução do sistema.

Por fim, todo o sistema desenvolvido está disponível em código aberto, incluindo os pesos da rede neural de segmentação, permitindo que este trabalho seja utilizado em aplicações em odometria visual, e que possa servir de base para muitos trabalhos a seguir.

Referências Bibliográficas

[Aharon, Orfaig e Bobrovsky 2022]AHARON, N.; ORFAIG, R.; BOBROVSKY, B.-Z. Bot-sort: Robust associations multi-pedestrian tracking. *arXiv preprint arXiv:2206.14651*, 2022.

[Badrinarayanan, Handa e Cipolla 2015]BADRINARAYANAN, V.; HANDA, A.; CIPOLLA, R. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. 05 2015.

[Bay, Tuytelaars e Gool 2006]BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. In: LEONARDIS, A.; BISCHOF, H.; PINZ, A. (Ed.). *Computer Vision – ECCV 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 404–417. ISBN 978-3-540-33833-8.

[Bescos et al. 2018]BESCOS, B. et al. Dynaslam: Tracking, mapping and inpainting in dynamic scenes. v. 3, p. 1–1, 07 2018.

[Bewley et al. 2016]BEWLEY, A. et al. Simple online and realtime tracking. In: *2016 IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2016. p. 3464–3468.

[Calonder et al. 2010]CALONDER, M. et al. Brief: Binary robust independent elementary features. In: . [S.l.: s.n.], 2010. v. 6314, p. 778–792. ISBN 978-3-642-15560-4.

[Chen et al. 2018]CHEN, L.-C. et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 40, n. 4, p. 834–848, 2018.

[Chen et al. 2022]CHEN, T. et al. Whuvid: A large-scale stereo-imu dataset for visual-inertial odometry and autonomous driving in chinese urban scenarios. *Remote Sensing*, v. 14, n. 9, 2022. ISSN 2072-4292. Disponível em: <https://www.mdpi.com/2072-4292/14/9/2033>.

[Chen et al. 2022]CHEN, Z. et al. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022.

[Chien et al. 2016]CHIEN, H.-J. et al. When to use what feature? sift, surf, orb, or a-kaze features for monocular visual odometry. In: *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. [S.l.: s.n.], 2016. p. 1–6.

[Csurka, Volpi e Chidlovskii 2023]CSURKA, G.; VOLPI, R.; CHIDLOVSKII, B. *Semantic Image Segmentation: Two Decades of Research*. 02 2023.

[Davison et al. 2007]DAVISON, A. J. et al. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 29, n. 6, p. 1052–1067, 2007.

[Dosovitskiy et al. 2021]DOSOVITSKIY, A. et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. Disponível em: <https://arxiv.org/abs/2010.11929>.

[Dosovitskiy et al. 2015]DOSOVITSKIY, A. et al. Flownet: Learning optical flow with convolutional networks. In: *IEEE International Conference on Computer Vision (ICCV)*. [s.n.], 2015. Disponível em: <http://lmb.informatik.uni-freiburg.de/Publications/2015/DFIB15>.

[Du et al. 2023]DU, Y. et al. Strongsort: Make deepsort great again. *IEEE Transactions on Multimedia*, IEEE, 2023.

[Farnebäck 2003]FARNEBäCK, G. Two-frame motion estimation based on polynomial expansion. In: . [S.l.: s.n.], 2003. v. 2749, p. 363–370. ISBN 978-3-540-40601-3.

[Fischler e Bolles 1981]FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 24, n. 6, p. 381–395, jun 1981. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/358669.358692>.

[Freda 2024]FREDA, L. *pyslam*. [S.l.]: GitHub, 2024. https://github.com/luigifreda/pyslam.

[Fu, Xia e Qiao 2021]FU, D.; XIA, H.; QIAO, Y. Monocular visual-inertial navigation for dynamic environment. *Remote Sensing*, v. 13, n. 9, 2021. ISSN 2072-4292. Disponível em: <https://www.mdpi.com/2072-4292/13/9/1610>.

[Geiger, Ziegler e Stiller 2011]GEIGER, A.; ZIEGLER, J.; STILLER, C. Stereoscan: Dense 3d reconstruction in real-time. In: *Intelligent Vehicles Symposium (IV)*. [S.l.: s.n.], 2011.

[Harris e Stephens 1988]HARRIS, C.; STEPHENS, M. A combined corner and edge detector. In: *Proceedings of the 4th Alvey Vision Conference*. [S.l.: s.n.], 1988. p. 147–151.

[Hartley e Zisserman 2003]HARTLEY, R.; ZISSERMAN, A. *Multiple View Geometry in Computer Vision*. 2. ed. New York, NY, USA: Cambridge University Press, 2003. ISBN 0521540518.

[He et al. 2022]HE, K. et al. Masked autoencoders are scalable vision learners. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2022. p. 15979–15988.

[He et al. 2017]HE, K. et al. Mask r-cnn. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017. p. 2980–2988.

[Horn e Schunck 1981]HORN, B.; SCHUNCK, B. Determining optical flow. *Artificial Intelligence*, v. 17, p. 185–203, 08 1981.

[Jung et al. 2021]JUNG, S. et al. Moving object detection with single moving camera and imu sensor using mask r-cnn instance image segmentation. *International Journal of Precision Engineering and Manufacturing*, v. 22, 05 2021.

[Kirillov et al. 2023]KIRILLOV, A. et al. Segment anything. *arXiv:2304.02643*, 2023.

[Krizhevsky, Sutskever e Hinton 2012]KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, v. 25, 01 2012.

[Kundu, Krishna e Sivaswamy 2009]KUNDU, A.; KRISHNA, K. M.; SIVASWAMY, J. Moving object detection by multi-view geometric techniques from a single camera mounted robot. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.: s.n.], 2009. p. 4306–4312.

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

[Levenberg 1944]LEVENBERG, K. A method for the solution of certain non – linear problems in least squares. *Quarterly of Applied Mathematics*, v. 2, p. 164–168, 1944. Disponível em: <https://api.semanticscholar.org/CorpusID:124308544>.

[Li 2017]LI, S. A review of feature detection and match algorithms for localization and mapping. *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, v. 231, n. 1, p. 012003, sep 2017. Disponível em: <https://dx.doi.org/10.1088/1757-899X/231/1/012003>.

[Lin et al. 2020]LIN, T.-Y. et al. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 42, n. 2, p. 318–327, 2020.

[Liu et al. 2021]LIU, Z. et al. Swin transformer: Hierarchical vision transformer using shifted windows. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2021.

[Liu et al. 2022]LIU, Z. et al. Video swin transformer. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2022. p. 3192–3201.

[Long, Shelhamer e Darrell 2015]LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015. p. 3431–3440.

[Lowe 1999]LOWE, D. G. Object recognition from local scale-invariant features. In: IEEE. *Proceedings of the seventh IEEE international conference on computer vision*. [S.l.], 1999. v. 2, p. 1150–1157.

[Lucas e Kanade 1981]LUCAS, B.; KANADE, T. An iterative image registration technique with an application to stereo vision (ijcai). In: . [S.l.: s.n.], 1981. v. 81.

[Maczyta, Bouthemy e Meur 2019]MACZYTA, L.; BOUTHEMY, P.; MEUR, O. L. Unsupervised motion saliency map estimation based on optical flow inpainting. In: *2019 IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2019. p. 4469–4473.

[Mayer et al. 2016]MAYER, N. et al. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. [s.n.], 2016. ArXiv:1512.02134. Disponível em: <http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16>.

[Migliore et al. 2011]MIGLIORE, D. et al. Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments. *ICRA09 Workshop on Safe navigation in open and dynamic environments Application to autonomous vehicles*, 04 2011.

[Mourikis e Roumeliotis 2007]MOURIKIS, A. I.; ROUMELIOTIS, S. I. A multi-state constraint kalman filter for vision-aided inertial navigation. In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. [S.l.: s.n.], 2007. p. 3565–3572.

[Muja e Lowe 2009]MUJA, M.; LOWE, D. G. Fast approximate nearest neighbors with automatic algorithm configuration. *International Conference on Computer Vision Theory and Applications*, 2009.

[Mur-Artal e Tardós 2017]MUR-ARTAL, R.; TARDÓS, J. D. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, v. 33, n. 5, p. 1255–1262, 2017.

[Mur-Artal e Tardós 2017]MUR-ARTAL, R.; TARDóS, J. D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, v. 33, n. 5, p. 1255–1262, 2017.

[Mur-Artal Raúl; Montiel e Tardós 2015]MUR-ARTAL RAÚL; MONTIEL, J. M. M.; TARDÓS, J. D. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, v. 31, n. 5, p. 1147–1163, 2015.

[Nam e Gon-Woo 2020]NAM, D. V.; GON-WOO, K. Robust stereo visual inertial navigation system based on multi-stage outlier removal in dynamic environments. *Sensors*, MDPI AG, v. 20, n. 10, p. 2922, May 2020. ISSN 1424-8220. Disponível em: <http://dx.doi.org/10.3390/s20102922>.

[Nister 2004]NISTER, D. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 26, n. 6, p. 756–770, 2004.

[Olson, Leonard e Teller 2006]OLSON, E.; LEONARD, J.; TELLER, S. Fast iterative alignment of pose graphs with poor initial estimates. In: . [S.l.: s.n.], 2006. p. 2262 – 2269.

[Qin, Li e Shen 2018]QIN, T.; LI, P.; SHEN, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, v. 34, n. 4, p. 1004–1020, 2018.

[Radford et al. 2021]RADFORD, A. et al. Learning transferable visual models from natural language supervision. In: *International Conference on Machine Learning*. [s.n.], 2021. Disponível em: <https://api.semanticscholar.org/CorpusID:231591445>.

[Ren et al. 2015]REN, S. et al. Faster R-CNN: Towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems (NIPS)*. [S.l.: s.n.], 2015.

[Ronneberger, Fischer e Brox 2015]RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: NAVAB, N. et al. (Ed.). *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Cham: Springer International Publishing, 2015. p. 234–241. ISBN 978-3-319-24574-4.

[Rosten e Drummond 2005]ROSTEN, E.; DRUMMOND, T. Fusing points and lines for high performance tracking. In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. [S.l.: s.n.], 2005. v. 2, p. 1508–1515 Vol. 2.

[Rublee et al. 2011]RUBLEE, E. et al. Orb: an efficient alternative to sift or surf. In: . [S.l.: s.n.], 2011. p. 2564–2571.

[Samadzadeh e Nickabadi 2022]SAMADZADEH, A.; NICKABADI, A. *SRVIO: Super Robust Visual Inertial Odometry for dynamic environments and challenging Loop-closure conditions*. 01 2022.

[Sarlin et al. 2020]SARLIN, P.-E. et al. SuperGlue: Learning feature matching with graph neural networks. In: *CVPR*. [S.l.: s.n.], 2020.

[Scaramuzza e Fraundorfer 2011]SCARAMUZZA, D.; FRAUNDORFER, F. Visual odometry [tutorial]. *IEEE Robotics & Automation Magazine*, v. 18, n. 4, p. 80–92, 2011.

[Sevilla-Lara et al. 2016]SEVILLA-LARA, L. et al. Optical flow with semantic segmentation and localized layers. In: . [S.l.: s.n.], 2016.

[Szeliski 2010]SZELISKI, R. Computer vision - algorithms and applications. In: ____. *Texts in Computer Science*. Second edition. [S.l.: s.n.], 2010. p. 109.

[Triggs et al. 2000]TRIGGS, B. et al. Bundle adjustment - a modern synthesis. *ICCV '99 Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, p. 198–372, 01 2000.

[Vaswani et al. 2017]VASWANI, A. et al. Attention is all you need. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017. (NIPS'17), p. 6000–6010. ISBN 9781510860964.

[Vertens, Valada e Burgard 2017]VERTENS, J.; VALADA, A.; BURGARD, W. Smsnet: Semantic motion segmentation using deep convolutional neural networks. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [S.l.: s.n.], 2017. p. 582–589.

[Wang et al. 2020]WANG, J. et al. Displacement-invariant matching cost learning for accurate optical flow estimation. *Advances in Neural Information Processing Systems*, v. 33, 2020.

[Wang et al. 2023]WANG, W. et al. Image as a foreign language: BEiT pretraining for vision and vision-language tasks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2023.

[Wang et al. 2023]WANG, W. et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2023. p. 14408–14419.

[Wojke, Bewley e Paulus 2017]WOJKE, N.; BEWLEY, A.; PAULUS, D. Simple online and realtime tracking with a deep association metric. In: IEEE. *2017 IEEE International Conference on Image Processing (ICIP)*. [S.l.], 2017. p. 3645–3649.

[Yan, Zhang e Wu 2022]YAN, H.; ZHANG, C.; WU, M. Lawin transformer: Improving semantic segmentation transformer with multi-scale representations via large window attention. *ArXiv*, abs/2201.01615, 2022. Disponível em: <https://api.semanticscholar.org/CorpusID:245704446>.

[Zhang et al. 2022]ZHANG, Y. et al. Bytetrack: Multi-object tracking by associating every detection box. 2022.