Odometria Visual Monocular

Lucas Reis das Chagas
Instituto de Informática
Universidade Federal de Goiás
Goiânia, Brasil
lucas234@discente.ufg.br

January 31, 2024

Resumo - A odometria visual monocular é uma técnica muito utilizada na robótica em que estima a posição e orientação de uma câmera ao longo do tempo utilizando frames consecutivos capturados por uma câmera monocular e sua matriz de parâmetros intrínsecos. Sabendo que o processo está sujeito a erros principalmente devido ao fato de estarmos utilizando coordenadas 2D para estimar coordenadas 3D, esse trabalho utiliza a ideia do pipeline convencional de odometria visual monocular para tentarmos contornar problemas como ambiguidade na escala, efeito de oclusão, erros de correspondência e, além disso, trarei ideias para diminuirmos o acúmulo de erros gerados ao longo do tempo.

I. Introdução

A odometria visual monocular é um campo estudado pela robótica e visão computacional que busca extrair informações relacionadas a movimento e orientação utilizando-se de sequências de imagens provenientes de uma única câmera.

Entre as vantagens de utilizar uma câmera para realizar odometria se destacam o fato de poder ser utilizada tanto em ambientes indoor quanto outdoor e o seu baixo custo, visto que podemos extrair movimento com apenas 1 câmera.

A abordagem utilizada nessa técnica tem como objetivo estimar movimento tridimensional tendo como entrada informações bidimensionais, portanto não temos como obter com exatidão, sem conhecimentos adicionais, informações como profundidade da cena, tendo que utilizar métodos que estimam essa dimensão.

O objetivo desse artigo é apresentar o pipeline convencional de odometria visual monocular e discutir a respeito de otimizações que retornam uma estimativa mais precisa.

Tendo em vista essa proposta, este artigo foi dividido em seções buscando oferecer uma melhor compreensão para o leitor. A seção II aborda sobre noções fundamentais que fazem parte dessa odometria, a seção III refere-se à metodologia empregada na estimativa do movimento e as seções IV e V remetem-se a apresentar resultados e conclusões obtidos.

II. Noções Fundamentais

Este capítulo foi dividido em seções a fim de segmentar o conteúdo a ser explicado, portanto, na tabela a seguir, estão os temas que serão abordados

Table 1: Temas a serem abordados

Index	Tema
A	Correspondência de pontos
B	Matriz Fundamental
C	RANSAC
D	Matriz Essencial
E	Triangulação

A - Correspondência de pontos

Para estimarmos a rotação e translação que ocorre entre 2 frames, inicialmente precisamos encontrar pontos correspondentes nas duas imagens a fim de verificar o quão eles variaram na cena.

Sabendo que em uma imagem existe inúmeras informações, temos que extrair elementos que sejam singulares (como pontos invariantes à rotação e escala, cantos, robustez à alterações de iluminação, etc) e, para isso, existem diversos métodos e implementações que podem ser usadas como SIFT, SURF, ORB, KLT entre outros. Devido ao fato de utilizar o SIFT (*Scale-Invariant Feature Transform*) na implementação do algoritmo que apresentará os resultados neste artigo, irei explicar a ideia por trás de sua implementação.

O SIFT é um algoritmo que identifica pontos de interesse baseados em duas etapas:

Detectores: Nesse estágio, busca-se encontrar pontos característicos, portanto o método utiliza diferenças de gaussianas com valores de sigmas diferentes (simulando um filtro laplaciano da gaussiana, pois nos retornará bordas presentes na nossa imagem) juntamente com pirâmides buscando encontrar pontos invariantes à escala. Após isso, é indentificado pontos chave e realizado um refinamento na localização deles.

Descritores: Para cada detector, é importante termos como diferenciá-lo dos outros, portanto é criado um descritor para esse ponto utilizando-se as magnitudes e orientações dos gradientes dentro de uma janela ao redor do ponto e criando um histograma de gradientes normalizados que descreve aquela região ao redor do ponto de interesse.

Após a captura dos detectores e descritores entre 2 frames, é utilizado medidas para realizar a comparação dos detectores entre os frames, como por exemplo KNN (*K-Nearest Neighbors*) juntamente com o critério de Lowe. Por fim, vale ressaltar que o SIFT, assim como outros algoritmos, possuem vantagens e desvantagens, sendo sua principal vantagem o fato

de ser bastante robusto quanto a captura dos detectores e descritores e sua principal desvantagem sendo seu alto custo computacional.

B - Matriz Fundamental

A matriz fundamental F é um conceito da geometria epipolar em que descreve as relações entre pelo menos duas vistas de uma mesma cena. Essa matriz é responsável por relacionar os pontos correspondentes e define uma linha epipolar *l'* (representada pelo produto vetorial do epipolo com o mapeamento de transferência de um ponto de uma imagem para outra (**H**))

$$l' = [\mathbf{e}]_\times \mathbf{H}\mathbf{x} = \mathbf{F}\mathbf{x}$$

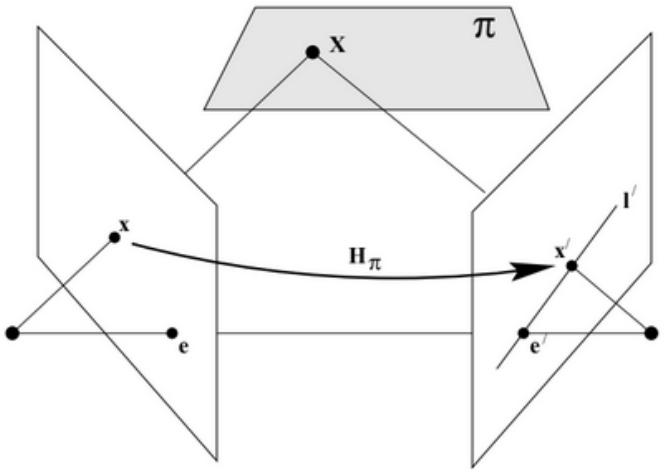

Figure 1: Geometria Epipolar. Fonte:[1]

Visto que a linha epipolar representa uma reta (nesse caso, uma reta na segunda imagem) em que estará o nosso ponto de correspondente, ao realizar o produto escalar desse ponto com a reta, encontraremos um valor nulo, portanto:

$$\mathbf{x'}^T \mathbf{F}\mathbf{x} = 0 \tag{1}$$

Como a nossa matriz fundamental $\mathbf{F}_{3x3}$ possui 9 elementos então, em tese, teríamos que possuir pelo menos 9 conjuntos de pontos correspondentes ${\mathbf{x'}^T}_{1x3}$ (vetor que representa o ponto correspondente em coordenada homogênea na segunda imagem) e $\mathbf{x}_{3x1}$ (vetor que representa o ponto correspondente em cooredenada homogênea na primeira imagem), porém podemos diminuir o grau de liberdade da matriz fundamental, visto que ela pode variar devido a um escalar, ou seja, para um escalar λ, ao multiplicá-lo pela equação (1), manterá a igualdade. Há também implementações em que se utilizam menos pontos, porém, na realização desse trabalho, foram utilizados 9 conjuntos de pontos correspondentes para compor inicialmente a matriz fundamental (representada em forma de vetor coluna $\mathbf{f}_{9x1}$) seguindo a equação a seguir:

$$\mathbf{A}\mathbf{f} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = 0 \tag{2}$$

C - RANSAC

O RANSAC *Random Sample Consensus* é um método iterativo que nos ajuda a estimar o melhor modelo matemático dentre os que ele seleciona aleatoriamente. No nosso caso, mesmo possuindo um método robusto como SIFT em que nos entrega bons pontos correspondentes, ao selecionar 9 conjunto de pontos desses retornados pelo SIFT, há a possibilidade de pegarmos escolhas ruins como pontos não espalhados pela cena (muito próximos, por exemplo) ou até mesmo pontos em que o SIFT possa ter classificado erroneamente, o que resultaria em um matriz fundamental "ruim".

Como normalmente temos um conjunto maior que 9 pontos correspondentes, ao selecionar a quantidade minima para montar a matriz fundamental, podemos testá-la para os demais pontos correspondes e, devido à equação (1), deveria encontrar 0 como resposta (ou algo próximo disso, devido às incertezas).

Portanto, o RANSAC será responsável por escolher aleatóriamente n grupos de 9 conjuntos de pontos correspondentes e testará a matriz **F** para de cada grupo escolhido e contabilizará a quantidade de respostas próximas de 0 (módulo da resposta menor que 0.15, por exemplo) que cada um dos grupos obteve, vencerá o grupo de pontos que obteve a maior quantidade respostas que satisfazem o limiar escolhido. Além disso, ao realizar o método RANSAC e encontrar pontos inliers da nossa matriz fundamental, podemos combiná-los com os 9 pontos que formaram a matriz a fim de gerar um nova matriz **F** mais precisa.

D - Matriz essencial

Sabendo que um ponto na imagem (2D) é uma projeção (com isso, consideramos parâmetros intrínsecos e extrínsecos da camera) de um ponto no mundo real (3D)

$$\mathbf{x} = \mathbf{P}\mathbf{X} \quad ; \quad \mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}] \tag{3}$$

$$\mathbf{x'} = \mathbf{P'}\mathbf{X} \quad ; \quad \mathbf{P'} = \mathbf{K'}[\mathbf{R}|\mathbf{t}] \tag{4}$$

podemos extrair um caso particular da matriz **F**, a matriz essencial, que seria a matriz fundamental quando consideramos os seus parâmetros intrínsecos como normalizados (representados pela matriz identidade *I*), portanto, ao encontrar a matriz essencial **E**, nós temos uma matriz que representa somente os parâmetros extrínsecos (rotação e translação da cena), que é o que buscamos. Para obter essa matriz, tendo a matriz fundamental, basta realizar:

$$\mathbf{E} = \mathbf{k'}^T \mathbf{F}\mathbf{k} \tag{5}$$

E - Triangulação

Triangulação é um método utilizado para determinar a posição de um ponto no espaço tridimensional a partir de pelo menos 2 projeções desse ponto em uma imagem. Na equação (3) por exemplo, vimos que $\mathbf{x} = \mathbf{P}\mathbf{X}$, como estamos nos referindo ao mesmo vetor, o produto vetorial desses elementos deve ser nulo, portanto: $\mathbf{x}_\times \mathbf{P}\mathbf{X} = 0$.

Com essa igualdade, nós encontraremos 3 equações, porém 1 é combinação linear das outras 2, porém precisamos estimar $\mathbf{X}$ que é um ponto do plano 3D (se contabilizarmos a coordenada homogênea, possui 4 dimensões), portanto utilizaremos a projeção do ponto 3D no outro frame $\mathbf{x'}_{\times}\mathbf{P'X} = 0$ e encontraremos a solução para $\mathbf{AX} = 0$, sendo:

$$A = \begin{bmatrix} x\mathbf{p}^{3T} - \mathbf{p}^{1T} \\ y\mathbf{p}^{3T} - \mathbf{p}^{2T} \\ x'\mathbf{p'}^{3T} - \mathbf{p'}^{1T} \\ y'\mathbf{p'}^{3T} - \mathbf{p'}^{2T} \end{bmatrix} \tag{6}$$

Quando extrairmos rotação e translação da matriz essencial, obteremos 2 possíveis rotações e 2 possíveis translações, o que contabiliza 4 combinações possíveis distribuidas da seguinte forma:

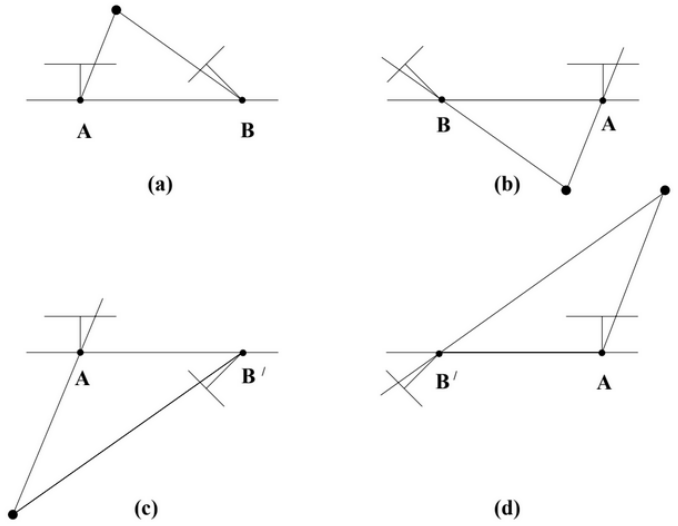

Figure 2: Combinações possíveis. Fonte:[1]

Entre essas combinações, nós buscamos escolher aquela em que o nosso ponto tridimensional se apresenta a frente da câmera (ou seja, a combinação (**a**)) o que significa dizer que ao realizarmos a triangulação verificaremos a quantidade de pontos em que possuem sua coordenada de profundidade (z no caso de coordenadas de câmera) positiva.

III. Metodologia

Inicialmente, devemos definir a nossa posição e orientação inicial juntamente com nosso sistema de coordenadas. Como esse artigo foi feito com base nos datasets KITTI, portanto não tendo um ponto de interesse para ser o meu sistema de coordenadas inicial, irei considerar a posição e orientação inicial sendo a do primeiro frame da seguinte forma:

Figure 3: Orientação inicial Fonte:[8]

Após definir o nosso ponto inicial, nosso objetivo passa a ser determinar qual foi o movimento (rotação e translação) realizado nos frames consecutivos, para isso, utilizaremos os conceitos explicados na *seção II*. Começaremos criando o nosso vetor de estado (vetor que guardará nossa posição e orientação ao longo do tempo), para esse trabalho, foi definido da seguinte forma:

$$X_{\text{state}} = \begin{bmatrix} 0 & \#\ \text{tx} \\ 0 & \#\ \text{ty} \\ 0 & \#\ \text{tz} \\ 0 & \#\ \theta_x \\ 0 & \#\ \theta_y \\ 0 & \#\ \theta_z \end{bmatrix}_{6\times 1}$$

Ao receber o primeiro frame, o primeiro passo é extrair os detectores e descritores da imagem, por exemplo, utilizando o método SIFT. Após isso, é necessário esperar pelo segundo frame, pois um único frame não fornece informações suficientes para uma estimativa precisa. Após extrair as mesmas informações da segunda imagem, devemos estabelecer matches entre detectores utilizando os descritores pois, assim, obteremos as correspondências. Para obter essas equivalências, pode-se ser utilizado o algoritmo KNN em que é possível comparar para cada ponto do primeiro frame quais seriam os dois pontos no segundo frame mais prováveis de serem considerados correspondentes e, com isso, é comparado a distância da melhor correspondência com a distância da segunda melhor correspondência e, caso essa distância seja aceitável para supormos unicidade entre o ponto com seu mais provável correspondente, realizamos um match entre esses pontos.

Tendo um conjunto de pontos correspondentes, podemos utilizar o método RANSAC juntamente com a equação (1) para tentar obter a matriz fundamental formada pelos melhores pontos correspondentes. Utilizar esse método é importante pois, além de ser robusto evitando outliers, faz com que possamos nos livrar de pontos não estáticos na cena, ou seja, pontos que, além da câmera, estão se movendo.

Após a obtenção da matriz fundamental, podemos encontrar a matriz essencial através da equação (5), note que como estamos utilizando uma única câmera durante o processo, temos a mesma matriz de parâmetros intrínsecos, ou seja, $\mathbf{k'} = \mathbf{k}$. Como a matriz essencial nos fornece informações sobre a rotação e translação da seguinte forma: $\mathbf{E} = [\mathbf{t}] \times \mathbf{R}$, podemos utilizar decomposição SVD para encontramos as matrizes que queremos. Ao realizar esses passos, encontraremos 4 combinações (mostradas na Figura 2) possíveis que satisfazem a matriz essencial, porém a combinação que nos interessa é aquela na qual o ponto projetado se encontra à frente da câmera.

Para cada combinação possível de rotação-translação teremos diferentes matrizes de projeção $\mathbf{P'}$ visto na equação (4), portanto, utilizando a matriz de projeção do ponto 3D do primeiro frame $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$, a matriz de projeção do ponto 3D do segundo frame $\mathbf{P} = \mathbf{K'}[\mathbf{R}|\mathbf{t}]$ e alguns pontos correspondentes, encontramos a matriz $\mathbf{A}$ através da triangulação (equação (6)) e, com isso, podemos estimar o ponto tridimensional $\mathbf{X}$ através do produto $\mathbf{AQ_1} = \mathbf{0}$.

A fim de encontrar a melhor combinação, ao se estimar os pontos 3D, devemos aplicar a transformação $\mathbf{T} = [\mathbf{R}|\mathbf{t}]$ ao nosso ponto $\mathbf{Q_1}$ obtendo, assim, um segundo ponto tridimensional transladado e rotacionado $\mathbf{Q_2} = \mathbf{TQ_1}$. Como dito anteriormente, a combinação rotação-translação que buscamos é aquela que nos retornará a maior quantidade de pontos 3D com profundidade (coordenada z em coordenadas de câmera) positiva, ou seja, para as 4 combinações iremos somar a quantidade de

pontos $\mathbf{Q_1}$ e $\mathbf{Q_2}$ com z positivo e escolher a que apresenta a maior quantidade.

Após encontrar as matrizes de rotação e translação, o próximo passo é tentar estimar a escala. Visto que a odometria visual monocular não nos permite encontrar a escala absoluta da cena, sem informações adicionais, foi optado por encontrar uma escala relativa a fim de servir de escala para nossa matriz de translação somente (devido ao fato da matriz de rotação não sofrer muita variação em relação à escala), na qual essa escala relativa se relaciona com os pontos $\mathbf{Q_1}$ e $\mathbf{Q_2}$ da seguinte forma:

$$\text{escala relativa} = \frac{1}{N}\sum_{i=1}^{N}\frac{\|Q1_i - Q1_{i+1}\|}{\|Q2_i - Q2_{i+1}\|} \tag{7}$$

Assim que obtivermos as matrizes de rotação e translação, devemos atualizar o nosso vetor de estado, acrescentando nele as informações de variações angulares (por exemplo, decompondo a matriz de rotação em ângulos de Euler para achar $\theta_x, \theta_y, \theta_z$) e de translado. Ao tentar realizar essa parte devemos ficar atentos principalmente com 2 pontos:

Certifique-se de ajustar as coordenadas de câmera com as coordenadas que voce adotou inicialmente: Devido ao fato do sistema de coordenadas da câmera ser adotado diferentemente do sistema que costumamos adotar no cotidiano, talvez seja preciso realizar uma correspondência de coordenadas (por exemplo, usando a regra da mão direita), veja:

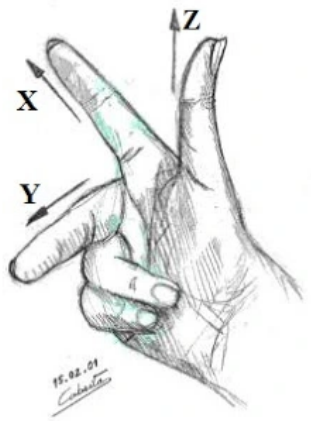

Figure 4:
Fonte: https://dukasatblog.wordpress.com/2019/03/16/ache-as-coordenadas/

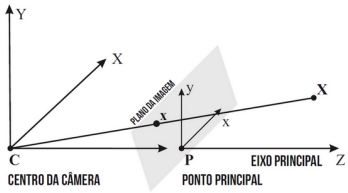

Figure 5: Fonte: ResearchGate (ver Rodapé 0)

Translação referente ao seu sistema de coordenadas de câmera: Quando encontrar a translação, ela representará movimento da camera em relação a posição anterior e não em relação ao seu sistema inicial. Felizmente, podemos utilizar a orientação presente no nosso vetor de estado para rotacionarmos a nossa traslação, fazendo assim com que ela corresponda ao nosso sistema de coordenadas inicial:

$$R_x \times R_y \times R_z \times V_n$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\text{sen}(\alpha) \\ 0 & \text{sen}(\alpha) & \cos(\alpha) \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta) & 0 & \text{sen}(\beta) \\ 0 & 1 & 0 \\ -\text{sen}(\beta) & 0 & \cos(\beta) \end{bmatrix} \cdot \begin{bmatrix} \cos(\chi) & -\text{sen}(\chi) & 0 \\ \text{sen}(\chi) & \cos(\chi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} \cos(\beta).\cos(\chi) & -\cos(\beta).\text{sen}(\chi) & \text{sen}(\beta) \\ \text{sen}(\alpha).\text{sen}(\beta).\cos(\chi)+\cos(\alpha).\text{sen}(\chi) & -\text{sen}(\alpha).\text{sen}(\beta).\text{sen}(\chi)+\cos(\alpha).\cos(\chi) & -\text{sen}(\alpha).\cos(\beta) \\ -\cos(\alpha).\text{sen}(\beta).\cos(\chi)+\text{sen}(\alpha).\text{sen}(\chi) & \cos(\alpha).\text{sen}(\beta).\text{sen}(\chi)+\text{sen}(\alpha).\cos(\chi) & \cos(\alpha).\cos(\beta) \end{bmatrix}}_{R_{xyz}} \cdot \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix}$$

Figure 6: Fonte:[7]

Atente-se ao fato de ter feito a correspondência dos sistemas de coordenadas anteriormente e, após isso, esse produto matricial retornará a translação que deve ser adicionada às três primeiras linhas do vetor de estado,e deverá ser adicionado os ângulos de Euler às três últimas linhas do vetor de estado.

Por fim, devem ser guardados os detectores e descritores do último frame para que o frame seguinte seja comparado com este.

IV. Experimentos e Resultados

Buscando apresentar resultados, foi criado um script python em que realiza essas etapas e testado com o dataset KITTI. Além disso, foram utilizadas algumas métricas apresentando comparações entre as posições obtidas através da odometria visual monocular e as posições fornecidas pelo KITTI. Para cada dataset escolhido, foram realizadas comparações entre as 250 primeiras poses. As métricas utilizadas foram:

Table 2: Métricas

Símbulo	Nome
AE	Erro médio
RMSE	Erro Médio Quadrático
RTE	Erro relativo de trajetória
ME	Erro Máximo
RV	Variação relativa

Os gráficos mostrados a seguir foram feitos considerando a trajetória vermelha como sendo o nosso ground truth, fornecidas pelo próprio KITTI, e a trajetória verde representa a nossa odometria visual monocular.

Primeiro Dataset:

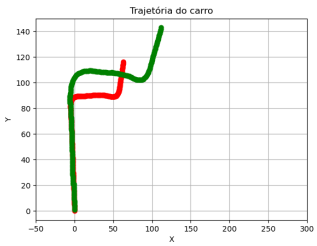

Figure 7

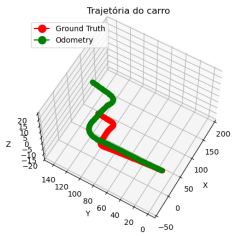

Figure 8

	x	y	z
AE	15.999529	12.336455	3.966199
RMSE	22.996220	14.602512	4.358621
RTE	15.602974	12.261463	4.035954
ME	48.950199	26.627214	6.371521
RV	1.585224	0.817951	0.323359

[0]URL: https://www.researchgate.net/figure/Figura-2-Representacao-do-sistema-de-coordenadas-no-modelo-Pinhole-Fonte-Adaptacao-de_fig2_298352094

Segundo Dataset:

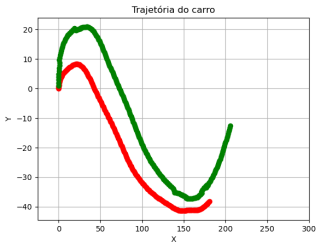

Figure 9

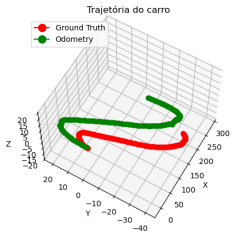

Figure 10

	x	y	z
AE	15.633919	8.792058	7.623112
RMSE	16.671636	10.118853	9.513446
RTE	15.746078	8.598048	7.768917
ME	24.703141	25.488619	16.093915
RV	1.166949	2.489913	1.287053

Terceiro Dataset:

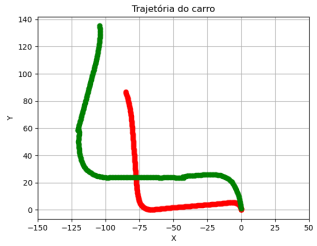

Figure 11

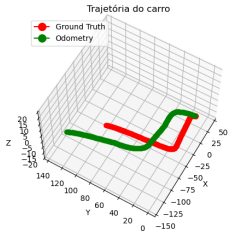

Figure 12

	x	y	z
AE	27.130710	28.761960	4.935562
RMSE	29.639156	30.917911	6.002114
RTE	27.835642	28.845624	4.876406
ME	43.547963	48.562704	13.337184
RV	0.736236	1.555818	17.945104

V. Conclusão

Como observado nos exemplos mostrados, conseguimos acertar a característica do movimento e obter uma boa rotação, mas ainda temos uma escala que talvez precise passar por um refinamento, o que já era esperado. Perceba que melhorias como aumentar a quantidade de matrizes fundamentais que serão testadas pelo método RANSAC podem serem feitas de acordo com a disponibilidade de recursos (por exemplo, computacionais) buscando evitar outliers.

Algo que pode ser feito para aumentar a precisão da nossa estimativa de pose é realizar uma fusão de informações vindas de outros sensores. Como conseguimos obter uma odometria vindo da câmera e de um IMU (*inertial measurement unit*), por exemplo, podemos unir essas informações através de filtros (como Kalman) e, assim, obter uma pose mais próxima da real.

Portanto, este trabalho realizou a implementação do pipeline convencional de odometria visual monocular explicando a teoria e passando por algumas ideias que são utilizadas para uma melhor estimativa de pose da câmera. Embora tenhamos alcançado resultados consideráveis em aspectos como rotação, reconhecemos que há margem para melhorias, particularmente na precisão da escala.

References

[1] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, mar 2004.

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd ed. Springer, 2022.

[3] M. C. dos Santos, *Revisão de Conceitos em Projeção, Homografia, Calibração de Câmera, Geometria Epipolar, Mapas de Profundidade e Varredura de Planos*. UNICAMP, Campinas, Outono/Inverno de 2012.

[4] H. M. Gräbin, *Odometria Visual Monocular: Estudo de caso para veículos terrestres*. Universidade Federal do Rio Grande do Sul, Escola de Engenharia, Departamento de Engenharia Elétrica, Porto Alegre, 2019.

[5] N. Nicolai. *Visual Odometry with Monocular Camera For Beginners: A Project in OpenCV*. YouTube, mar 2022. Disponível em: `https://www.youtube.com/watch?v=N451VeA8XRA&list=PL0aGtEdJ77pxr7MFFEUT4oOOatuZ9Upeo&index=37&t=743s`.

[6] Anônimo. *SIFT (Scale Invariant Feature Transform)*. Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Acessado em 2024.

[7] "Rotação em R3 e Projeção no Plano." Material didático. Geogebra. Acessado em 2024.

[8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "KITTI Odometry Dataset," [Online]. url: `https://www.cvlibs.net/datasets/kitti/eval_odometry.php`