Universidade Federal de Pernambuco
Centro de Informática

Bacharelado em Ciência da Computação

Um método para avaliação de mapas de disparidade utilizando técnicas de Aprendizagem de Máquina

Alana Laryssa Seabra de Albuquerque Santos

Trabalho de Graduação

Recife, Julho de 2018

Universidade Federal de Pernambuco
Centro de Informática

Um método para avaliação de mapas de disparidade utilizando técnicas de Aprendizagem de Máquina

Trabalho apresentado ao Programa de Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em em Ciência da Computação

Aluna: Alana Laryssa Seabra de Albuquerque Santos
Orientador: George Darmiton da Cunha Cavalcanti

Recife, Julho de 2018

Resumo

A reconstrução 3D através da Visão Estéreo tem sido um desafio muito importante em Visão Computacional há anos. Ultimamente têm-se dado ainda mais visibilidade devido à popularização de aplicações como jogos, drones, realidade virtual e navegação autônoma. Este problema, apesar de muito pesquisado na academia, ainda não foi considerado resolvido devido a sua alta complexidade e pouca capacidade de generalização. Apesar disso, por causa de seu baixo custo e resultados competitivos, a abordagem utilizando Visão Estéreo continua sendo bastante utilizada.

A principal dificuldade da Visão Estéreo é o mapeamento dos pontos correspondentes entre uma imagem e outra - conhecido como o problema da correspondência estéreo. A distância entre pixels congruentes é conhecida como disparidade. Para avaliar e realizar *benchmarking* de possíveis algoritmos que solucionam este problema, é necessário conhecer os reais valores de disparidade dos pontos das imagens. No entanto, a obtenção deste *ground-truth* é uma tarefa complexa e necessita de outros aparelhos para a coleta dos dados.

Ainda assim, a existência de medidas de qualidade para possíveis soluções para o problema da correspondência estéreo é de alta relevância. Este trabalho utiliza-se de técnicas de Aprendizagem de Máquina para o desenvolvimento de uma medida para avaliação de algoritmos de estimação de mapa de disparidade robusta à ausência de dados de *ground-truth*. Os dados de treinamento são os pixels do mapa de disparidade gerado pelo algoritmo sob os pares de imagem do banco de dados. Utilizou-se do *ground-truth* para rotular se a disparidade foi estimada de maneira correta ou errada. Uma vez concluída a fase de treinamento, podem ser testados os mapas de disparidade estimados pelo mesmo algoritmo sob quaisquer pares de imagens estéreo. A saída indica se o valor do pixel estimado está correto ou não.

Os resultados mostram a performance de diversos classificadores operando sobre este tipo de dado, levando em consideração a acurácia do modelo, bem como o custo computacional exigido por ele. Utilizando o classificador Random Forest, foi possível obter uma medida de f1 de quase 85%, mantendo o tempo de execução bastante competitivo. Os valores obtidos comprovam a hipótese de que diversas áreas da Computação - neste caso, a Visão Computacional - têm muito a se beneficiar com a utilização de técnicas de Aprendizagem de Máquina.

Abstract

Reconstructing a 3D scenario using Stereo Vision has been a serious challenge in the field of Computer Vision for years. In recent times, the issue has been in spotlight because of applications such as games, drones, virtual and augmented reality and autonomous navigation. Although highly researched, this problem has not been considered solved due to its complexity and low generalization capacity. Nevertheless, the Stereo Vision approach is still very popular because of its low cost and competitive results.

Mapping correspondent points between the pair of images - in other words, stereo matching - is the bottleneck of the Stereo Vision pipeline. The distance between correspondent pixels is known as disparity. In order to evaluate and benchmark algorithms that try to solve this problem, knowing the real disparity values is essential. However, acquiring this kind of data is not a simple job and requires extra equipments.

Even so, the presence of quality measures for possible solutions of this problem is highly relevant. This work takes advantage of Machine Learning techniques in order to develop an evaluation measure for disparity maps estimation algorithms that is robust to the absence of ground-truth data. The disparities estimated by the algorithm for the dataset stereo images are the training data. The ground truth was used to label if the estimated disparity is right or wrong. Once the training phase is finished, tests can be done with disparity maps generated by the same algorithm over any pair of stereo images. The output points out if the the pixel value was correctly estimated or not.

The results show the performance of several classifiers considering the accuracy of the model, as well as the computational cost of each method. The Random Forest classifier achieved the best general f1 measure, reaching almost 85%, while keeping the run time very competitive. Overall the results obtained confirm the hypothesis that many Computer Science fields - in this case, Computer Vision - have a lot to gain by using Machine Learning techniques.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

Apesar da alta qualidade que uma imagem pode alcançar devido às câmeras dos dias atuais, ainda não é possível realizar diretamente a inferência de informações tais qual a distância de um objeto à câmera. Esta distância é também conhecida como profundidade. Conhecer a profundidade dos pontos de um ambiente é um problema fundamental da Visão Computacional, pois é imprescindível para a reconstrução 3D da cena. Usando técnicas de reconstrução 3D, pode-se determinar o perfil 3D de qualquer objeto e ainda conhecer as coordenadas de todos os pontos. Na indústria, estas informações são essenciais em áreas como Navegação Autônoma, Jogos, Cinema, Realidade Virtual e até em Medicina.

Atualmente, existem diversas maneiras de obter informações de profundidade de um ambiente, por exemplo, utilizando sensores de movimento como o Kinect (Izadi et al. 2011), *rangefinders*, sensores de ultrassom, entre outros que variam de acordo com o grau de precisão e o custo. Uma das soluções mais populares - e baratas - para este tipo de problema é o uso da Visão Estéreo.

A Visão Estéreo se fundamenta na visão binocular humana, de forma que os olhos são substituídos por câmeras e o nosso cérebro é representado por um *software* - que é responsável por colher as informações dos dois pontos de vista, processá-las e inferir informações de profundidade. Entretanto, esta tarefa não é natural para computadores do jeito que é para humanos. Grande parte dos algoritmos encontrados na literatura exigem um alto custo de processamento, além de baixa capacidade de generalização, ou seja, eles são sensíveis à alteração como variações de iluminação do cenário, imagens não-retificadas, qualidade, posicionamento, configurações das câmeras, etc Uma avaliação extensa desses algoritmos pode ser encontrada em (Scharstein & Szeliski 2002).

A configuração mais simples de um sistema de visão estéreo está ilustrada na Figura 1.1. O problema da estimação da profundidade neste tipo de sistema se resume em encontrar o pixel em cada imagem que correspondem ao mesmo ponto no mundo real. Este par de pixels é chamado de correspondência estéreo e a diferença de coordenadas entre eles é conhecida

como disparidade, cujo valor pode ser encontrado realizando operações de semelhança de triângulos. Desta forma, o valor da disparidade é inversamente proporcional à profundidade.

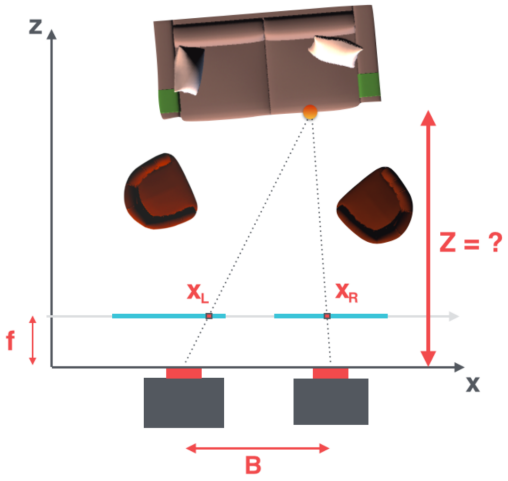


Figura 1.1: **Sistema de câmeras estéreo.** Duas câmeras, de distâncias focais f, separadas por uma distância B uma da outra. Ambas estão alinhadas em um mesmo eixo e apontadas para uma mesma direção. O par de pixels (x_L, x_R) é uma correspondência estéreo pois representam o mesmo ponto no mundo real. O problema consiste em calcular qual a distância Z (profundidade) deste ponto para as câmeras.

Por ser um problema inverso - tipo de problema cuja resolução é descobrir, a partir dos dados, os parâmetros de um modelo -, um algoritmo de estimação da profundidade dos pontos de um ambiente está suscetível a diversos tipos de erros. O mapeamento dos pixels das imagens se torna uma tarefa complicada principalmente devido a oclusões, objetos sem textura definida e estruturas repetidas. Devido à alta sensibilidade a erros, a avaliação da qualidade desses algoritmos é muito importante visto que pequenas variações no valor de disparidade de um ponto podem ocasionar um impacto significante na reconstrução 3D final.

Diversas medidas de confiança para as correspondências estéreo foram propostas na literatura (Hu & Mordohai 2012). Entretanto, elas dependem da existência de dados de *ground-truth* para comparação. A proposta deste trabalho é apresentar uma medida de avaliação de algoritmos de estimação de disparidade na ausência de *ground-truth* utilizando técnicas de Aprendizagem de Máquina.

1.2 Objetivo

Visto que as causas comuns dos erros de algoritmos de estimação de disparidade são conhecidas e bem documentadas, espera-se que métodos de aprendizagem sejam capazes de aprender e detectar alguns desses fatores.

Utilizando uma base de imagens estéreo com *ground-truths* de disparidade, o objetivo deste trabalho é utilizar o resultado de um algoritmo de estimação de disparidade aplicado a um par de imagens, compará-lo com o *ground-truth* associado e treinar um modelo de aprendizagem supervisionada baseado na acurácia do resultado do algoritmo.

Dado um novo mapa de disparidade sem *ground-truth* gerado pelo mesmo algoritmo, o modelo deve classificar cada pixel do mapa como correto ou errado. O modelo será responsável por aprender as características de um pixel com valor de disparidade correta e classificar os demais adequadamente.

1.3 Estrutura do trabalho

Este trabalho desdobra-se em 5 capítulos, incluindo este que introduz o problema a ser estudado e descreve o objetivo do documento. O Capítulo 2 revisa a literatura necessária para a fundamentação deste problema e define os conceitos básicos da Geometria Epipolar, da Visão Estéreo e da Aprendizagem de Máquina. O Capítulo 3 apresenta a metodologia utilizada para o desenvolvimento do trabalho, incluindo a análise da base de dados utilizada, o pré-processamento aplicado sobre ela e o modelo de aprendizagem. Os experimentos realizados e os resultados obtidos se encontram no Capítulo 4. O Capítulo 5 conclui este documento e sugere possíveis trabalhos futuros.

Capítulo 2

Fundamentação

Este capítulo apresenta os conceitos básicos necessários para completo entendimento do trabalho desenvolvido aqui. Primeiramente, ele irá introduzir brevemente as noções da geometria epipolar, que compoõem os princÌpios da *Multiple View Geometry* (Hartley & Zisserman 2003). Em seguida serão apresentadas as ideias fundamentais da visão estéreo, e, por fim, alguns conceitos de aprendizagem de máquina.

2.1 Geometria epipolar

A geometria epipolar é a geometria projetiva entre dois pontos de vista de uma mesma cena. é totalmente independente da estrutura da cena, sendo vinculada apenas aos parâmetros internos da câmera e a pose relativa entre elas. Ela é nomeada desta forma pois a linha que contém os centros óticos das câmeras, também conhecida como baseline, intersecta os planos imagens em pontos denominados epipolos.

Supondo um ponto X no espaço tridimensional e tomando x e x' como as projeções deste mesmo ponto nas primeira e na segunda imagens, respectivamente, o plano definido por estes 3 pontos é denominado plano epipolar. Devido às projeções necessárias para formação da imagem, os centros das câmeras também estão neste plano. Este plano epipolar corta os planos das imagens em linhas que são denominadas linhas epipolares, que são fundamentais para resolução dos desafios da visão estéreo.

A conceito principal da geometria epipolar é que, pra qualquer ponto X no espaço 3D, a relação $x'^T F x = 0$ é sempre satisfeita. Onde x e x' são as projeções do mesmo ponto X na primeira e na segunda visão, respectivamente. A matriz 3x3 F é conhecida como a matriz fundamental. Ela é, basicamente, a representação algébrica da geometria epipolar.

A matriz fundamental é derivada do mapeamento entre pontos e suas linhas epipolares correspondentes. Os detalhes desta derivação não serão explorados neste trabalho, portanto não serão descritos aqui. O que se deve ser ressaltado, no entanto, é o fato de que, utilizando a matriz fundamental, pode-se saber, por exemplo, em que linha sobre a imagem direita deve ser procurado o ponto correspondente a um ponto dado na imagem da esquerda.

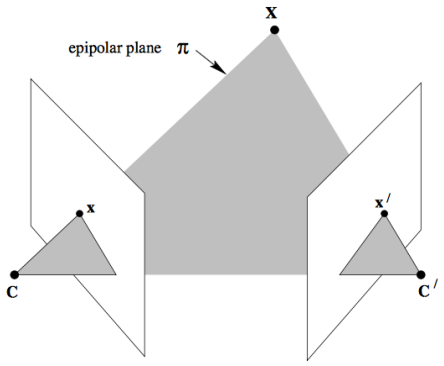


Figura 2.1: **Geometria epipolar** C e C' são os respectivos centros das câmeras esquerda e direita, o plano π é o plano epipolar e as linhas que são interseções do plano epipolar com os planos das imagens e contém x e x' são as linhas epipolares

2.2 Visão estéreo

2.2.1 Motivação

A projeção de raios de luz na nossa retina fornece ao nosso sistema visual uma imagem do mundo que é inerentemente bidimensional, e, mesmo assim, somos capazes de interagir com o mundo tridimensional como ele é. Até mesmo em situações que não conhecemos ou com objetos novos. Pode-se inferir, desse conhecimento, que uma das funções do sistema de visão humano é reconstruir uma representação 3D do mundo a partir de sua projeção bidimensional, de forma a oferecer uma sensação de profundidade.

O conceito base por trás da visão estéreo é, basicamente, o mesmo encontrado no sistema de visão humano, onde os nossos olhos esquerdo e direito serão substituídos por duas câmeras horizontalmente deslocadas e um software será responsável pelo processamento destas imagens, funcionando como o nosso cérebro.

Quando se fala de visão estéreo, estão inclusas duas maneiras de obter as imagens: ao mesmo tempo com dois receptores diferentes ou sequencialmente, por exemplo, com uma câmera que se move relativamente à cena. Estas duas situações são geometricamente equivalentes, portanto o sistema proposto é independente ao modo estéreo de aquisição das imagens. é importante frisar que, para este caso, as matrizes de câmeras, e consequentemente a matriz fundamental são dadas.

O principal objetivo da visão estéreo é encontrar a posição no espaço tridimensional de um ponto X, dadas suas imagens x e x' obtidas por duas câmeras diferentes e suas informações. Nestas circunstâncias, apenas uma projeção reversa dos raios das imagens não seria suficiente, pois geralmente os raios não se intersectam para formar o ponto exato por causa dos erros acumulados nas projeções x e x' da imagem. Portanto, é necessário estimar a melhor solução para o ponto tridimensional.

Através do método conhecido como triangulação, pode-se inferir o ponto X do espaço tridimensional, assumindo que os pontos correspondentes (ou homólogos) foram encontrados nas duas imagens. Este último problema é o mais crítico nas aplicações de visão estéreo, pois não é matematicamente fechado e está sujeito a uma variedade de erros.

2.2.2 Problema da correspondência de pontos

Para resolver o problema da correspondência de pontos, é preciso relacionar os pontos de uma imagem a pontos homólogos na outra imagem. De forma simplória, pode-se pensar em uma abordagem de força bruta e, pra cada ponto se uma imagem, procurar seu equivalente em todos os pontos da outra imagem. Entretanto, graças às restrições epipolares, pode-se reduzir o espaço de busca bidimensional para unidimensional. A restrição epipolar determina que a correspondência de um ponto x na primeira imagem está localizado na linha epipolar dele.

Uma vez que se sabe que o espaço de busca é unidimensional, se pode aplicar uma técnica de retificação nas imagens. A retificação transforma as imagens em uma configuração mais conveniente para o problema, permitindo que os pontos correspondentes se encontrem em um mesmo scanline, portanto, na mesma coordenada y.

2.2.3 Mapas de disparidade

A abordagem mais comum de resolver o problema das correspondências é através do cálculo do mapa de disparidade. Disparidade é a diferença entre a coordenada x de dois pontos homólogos, ou seja, $d = x - x'$. O valor da disparidade se relaciona inversamente com o valor da profundidade. Ou seja, uma vez sabida a disparidade, se pode calcular o valor da profundidade utilizando a distância focal f das câmeras e a distância entre os centros das câmeras, ou baseline, B na equação

$$Z = \frac{B * f}{d} \tag{2.1}$$

Existem diversos métodos de geração de mapa de disparidade [refs]. A ideia mais simplista é para cada linha da imagem esquerda, procurar o pixel mais semelhante na mesma linha da imagem direita. Um pixel sozinho não contém muita informação, por isso, geralmente, os métodos envolvem a medição de informações de uma janela em torno dele. As janelas são comparadas utilizando alguma função de similaridade definida por cada algoritmo, por exemplo, SSD, NCC, SAD, etc. O pixel candidato com maior similaridade é escolhido e sa distância entre eles é usada como a disparidade.

Mesmo em imagens retificadas, no entanto, o custo computacional de um método estimador de mapa de disparidade é alto. Grande parte dos métodos existentes não são passíveis de serem usados em aplicações que exigem processamento em tempo real. Os principais desafios que estes métodos enfrentam são as oclusões parciais ou totais de algum objeto na cena, padrões repetitivos, descontinuidade e distorções projetivas. Por este motivo, os algoritmos para tal finalidade costumam ser pouco generalizáveis, ou seja, eles são sensíveis a variações de iluminação do cenário, qualidade e configurações das câmeras, distorções nas imagens, etc. Este tema, portanto, é um dos mais pesquisados no universo da visão computacional. Revisões gerais e taxonomias sobre o tema podem ser encontradas em (Scharstein

& Szeliski 2002) e (Scharstein 1999).

Nos capítulo referente à metodologia, serão discutidos brevemente alguns algoritmos utilizados no projeto.

2.3 Aprendizagem de Máquina

De maneira resumida, Aprendizagem de Máquina pode ser definida como uma área de estudo que dá aos computadores a habilidade de aprender sem serem explicitamente programados. Este subcampo da Ciência da Computação evoluiu do estudo de reconhecimento de padrões e da teoria do aprendizado computacional. Apesar de não ser uma ciência nova, na última década têm ganhado um impulso mais significativo uma vez que o processamento computacional está cada vez mais barato e mais poderoso. Além disso, o crescente volume e variedade de dados disponíveis atualmente exigem modelos que permitam analisar dados grandes e complexos e fornecer resultados rápidos e precisos.

Em outras palavras, este campo estuda métodos de análise de dados que automatiza o desenvolvimento de modelos analíticos, permitindo que uma máquina aprenda interativamente a partir de dados e encontre insights sem ser explicitamente programada para procurá-los especificamente. E, embora seja uma uma subárea da Inteligência Artificial, a aprendizagem de máquina se intercepta amplamente com outras áreas do conhecimento, tais como estatísta, biologia, física, teoria da computação (Mitchell 1997).

A aprendizagem de máquina se divide em três categorias amplas, se acordo com a natureza do dado de aprendizado disponível. São elas:

1. **Aprendizado supervisionado:** o exemplos de entradas apresentados têm a saída desejada conhecida.

2. **Aprendizado não-supervisionado:** os dados de entrada não possuem rótulo, ou seja, as saídas desejadas são desconhecidas.

3. **Aprendizado por reforço:** o algoritmo descobre através de tentativa e erro quais ações geram as melhores feedbacks.

Neste trabalho, será usada apenas a abordagem supervisionada.

2.3.1 Aprendizagem supervisionada

Algoritmos de aprendizagem supervisionada realizam predições baseados em um conjunto de exemplo. Cada exemplo utilizado para treinamento é rotulado com o seu valor associado. Um algoritmo desta abordagem procura por padrões nestes rótulos, a fim de usá-los para inferir os rótulos dos dados de testes, que são desconhecidos.

Ou seja, dado um espaço de entrada X e seus rótulos, o espaço de saída Y, os algoritmos de aprendizagem supervisionada objetivam encontrar um mapeamento f que associa cada

entrada em X à sua respectiva saída em Y, satisfazendo $f : X- > Y$. Na maioria das aplicações, um valor no espaço de entrada X é um vetor d-dimensional, que pode ser formado por números ou não. Estes vetores são chamados de vetor de características e podem ser representações até de sinais, imagens, sons, etc. Similarmente, um valor no espaço de saída Y geralmente é representado por uma categoria de um conjunto finito de categorias ou um escalar de valor real. Neste primeiro caso, o problema é conhecido como classificação. No segundo, ele é chamado de regressão.

2.3.2 Classificação

A classificação é o problema da aprendizagem de máquina e da estatística que visa analisar uma nova instância do dado e identificar a qual categoria de um conjunto finito ela pertence. Esta análise se baseia no conjunto de dados de treinamento contendo exemplos no qual as categorias associadas são conhecidas. Um algoritmo que realiza esse tipo de trabalho é denominado classificador.

Por vezes a parte mais difícil de resolver um problema de aprendizagem de máquina é escolher o classificador que performe melhor dadas as circunstâncias do seu problema. Por ser um desafio bastante popular, são inúmeros os algoritmos propostos para resolver problemas de classificação. Tipos diferentes de dados de entrada e de saída e requisitos de performance do algoritmo são algumas das variáveis que precisam ser levadas em consideração.

As subseções abaixo apresentam os algoritmos de aprendizagem de máquina que serão utilizados ao longo dos experimentos deste trabalho.

K-Nearest Neighbors (KNN)

O KNN é um dos classificadores mais simples e um dos mais utilizados em Aprendizagem de Máquina. Ademais, é um algoritmo não paramétrico - o que significa que ele não assume que os dados pertecem a nenhuma distribuição particular. Ou seja, a estrutura do modelo é definida a partir dos dados. O funcionamento do KNN utiliza-se do espaço vetorial multidimensional de atributos dos dados e da distância entre cada ponto.

Por ser um algoritmo baseado em instância, ele não se preocupa em descobrir uma função que mapeie os exemplos observados às suas respectivas classes. Logo, a sua fase de treinamento consiste apenas no armazenamento dos vetores que representam os dados de treinamento e suas respectivas classes. Na fase de teste, o algoritmo recebe uma nova instância e calcula a distância entre ela e todas as instâncias de treinamento. A função de distância pode ser, por exemplo, a distância Euclidiana entre os pontos. Os K vizinhos mais próximos da nova instância irão determinar sua classe através do voto majoritário.

Além de não fazer suposições sobre o dado, ser de simples implementação e não ter custos de treinamento, o KNN lida com classificação com multiclasses de forma natural e seus resultados são competitivos. Por outro lado, ele pode ter alto custo computacional, devido à necessidade de guardar todo dado de treinamento e realizar todas os cálculos de

distância sempre que se testa uma nova instância. Outro fator negativo é que ele é sensível à características que não são muito relevantes, à escala do dado e a outliers.

Naive Bayes

O Naive Bayes é um método de classificação que engloba um grupo de simples classificadores probabilísticos que são fundamentados na aplicação do Teorema de Bayes. O termo *naive* existe devido à suposição que esta técnica faz de que os atributos dos dados são independentes. A atribuição de uma nova instância à sua classe é baseada na probablidade desse exemplo pertencer a essa classe.

Na fase de treinamento, o método estima os parâmetros de uma distribuição de probabilidade usando os dados de treinamento. Na fase de teste, para cada instância nova, o método computa a probabilidade a posteriori de aquele exemplo pertencer a cada classe do problema. Em seguida, a instância de teste é classificada de acordo com a maior probabilidade a posteriori encontrada.

Apesar de assumir independentes os atributos dos dados, na prática, algumas implementações do Naive Bayes têm se mostrado também eficiente quando essa suposição é falsa. Este método é uma boa escolha quando o usuário precisa não somente da predição da categoria, mas também o grau de certeza. Apesar de necessitar de pouco dado de treinamento, o algoritmo não se comporta bem se uma variável categórica tem uma categoria que não foi observada no conjunto de treinamento, pois o modelo estimará probabilidade zero.

Árvores de decisão

Ávores de decisão são uma forma simples, porém poderosa, de realizar uma análise de múltiplas variáveis. O objetivo a construção de um modelo que prediz a classe de uma instância baseado em várias perguntas realizadas às características desta instância - perguntas estas que são inferidas do próprio conjunto de treinamento. Uma árvore de decisão é desenhada de ponta cabeça com a sua raiz no topo. Cada nó da árvore representa uma condição e, a partir deles, são originadas as arestas, que são os ramos da árvore.

Uma das principais vantagens desta técnica é a facilidade de interpretação dos resultados, pois eles podem ser facilmente visualizados. Ademais, o método exige pouco pré processamento dos dados, ou seja, não é necessário realizar normalizações, preenchimento de variáveis, etc. Na prática, árvores de decisões têm boa performance até quando as regras inferidas não são integralmente condizentes com o modelo com o qual o dado foi gerado. Por outro lado, a técnica está sujeita a realizar um overfitting dos dados, além de poder ser instável - no sentido de ser sensível à pequenas mudanças nos dados originais.

Random Forest

Random Forest é um tipo de classificador supervisionado que utiliza a técnica de aprendizado por agrupamento. O algoritmo, em sua fase de treinamento, cria uma floresta através do agrupamento de várias árvores de decisão e o resultado dado na fase de teste é a combinação das previsões destas árvores. O uso dessa técnica é motivado pelo fato de que, quando combinadas decisões diferentes e independentes, sendo cada uma delas mais precisa do que um chute, os erros aleatórios se cancelam e as decisões corretas são reforçadas.

O objetivo principal deste modelo é colher diferentes decisões de pontos de vista aleatórios sobre os dados - a a aleatoriedade, nesse algoritmo, é aplicada em relação tanto às características, como às instâncias de treinamento. Dessa forma, o Random Forest tende a ter uma maior resistência às mudanças nos dados e ao ruído.

A limitação principal do classificador Random Forest é que ele pode se tornar lento quando usada uma quantidade grande de árvores - o que o torna impraticável para aplicações em tempo real. Suas vantagens incluem a facilidade de observar a importância relativa de cada característica dos dados e os seus parâmetros - que não são muitos e são de fácil interpretação. Além disso, o modelo dificulta a presença do overfitting, diferente das árvores de decisão.

Support Vector Machine (SVM)

Este algoritmo tem como objetivo encontrar a fronteira que separa as classes. Ele representa os exemplos como pontos no espaço cuja dimensão é igual ao número de características dos dados. Em seguida, o modelo mapeia os exemplos de forma que as instâncias que estão associadas à cada categoria sejam divididas por um espaço claro, que seja tão amplo quando possível. Ou seja, o algoritmo tenta encontrar, no espaço, um hiperplano que separe as classes, de maneira que esse hiperplano tenha a maior distância possível para os exemplos observados.

Por ser um algoritmo bastante complexo, o SVM tem boa performance em espaços de alta dimensão, além de ter bons resultados com diferentes tipos de dados - ou seja, boa capacidade de generalização. Para o algoritmo atingir uma alta performance, entretanto, as instâncias das classes não podem ser muito sobrepostas, nem podem possuir uma quantidade elevada de outliers. Além disso, a visualização e interpretação do modelo final não é simples e ele pode exigir bastante poder computacional se a quantidade dos dados for grande.

2.4 Análise de Componentes Principais (PCA)

O PCA É uma técnica do campo de Estatística que é utilizada para analisar inter-relações entre um grande número de variáveis e explicar estas variáveis em termos de suas componentes. O objetivo principal é encontrar um meio de resumir a informação contida nestas

variáveis de modo a perder o menor número de informações possível. O procedimento reorganiza os dados, projetando-os em outro espaço, nos quais os novos eixos são as componentes principais. Esta transformação é definida de forma que o primeiro componente principal tem a maior variância nos dados possível, e, cada componente seguinte respeita a restrição de ortogonalidade em relação ao eixo anterior e possui variância decrescente. Por causa de sua propriedade de encontrar os eixos que possuem maior informação sobre o dado, o PCA é uma técnica bastante utilizada para redução de dimensionalidade. (falar mais um pouco sobre)

Capítulo 3

Metodologia

Pode-se dividir o sistema proposto em 3 etapas. A primeira etapa, chamada de pré-processamento, gera os mapas de disparidade das imagens do banco de dados, rotula os pixels e formata os dados para serem compatíveis com a entrada da próxima etapa, que é a de aprendizagem. Nesta etapa, um modelo será treinado para aprender as características dos pixels do mapa de disparidade. Por fim, a última etapa, de avaliação, consiste em utilizar o modelo treinado para avaliar qualquer mapa de disparidade gerado pelo algoritmo selecionado.

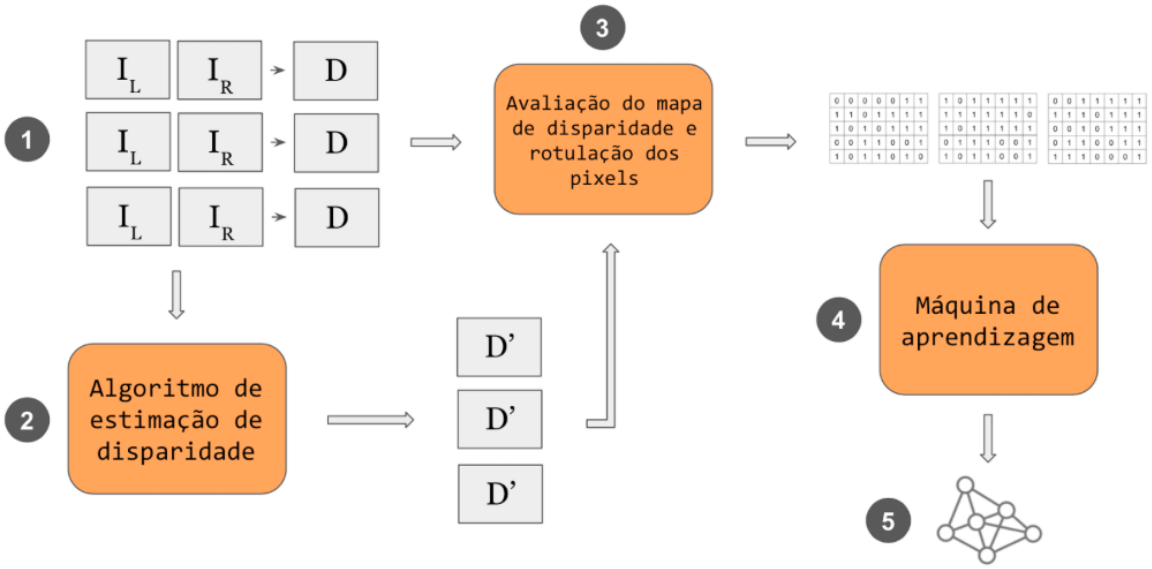


Figura 3.1: **Arquitetura da fase de treinamento do sistema.** I_L e I_R representam os pares de imagens fornecidas pelo banco de dados. D é o *ground-truth* de disparidade associado ao par. D' representa o mapa de disparidade resultante do algoritmo do passo 2.

A Figura 3.1 representa uma visão geral da fase de treinamento do sistema, enquanto a Figura 3.2 representa a visão da fase de teste. As variáveis mais relevantes do sistema - que

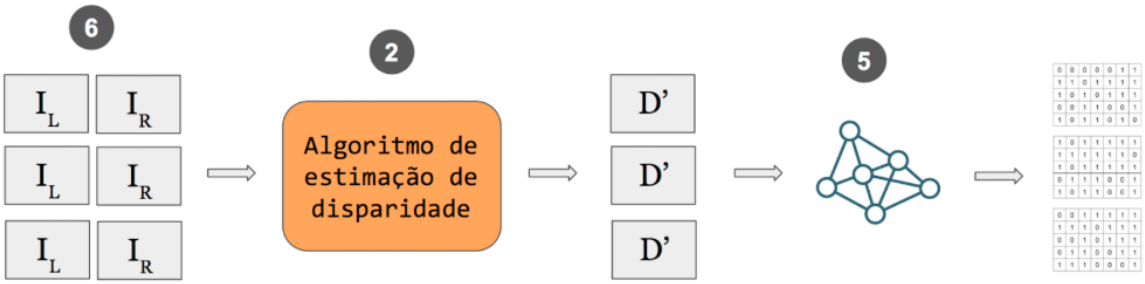


Figura 3.2: **Arquitetura da fase de testes do sistema.**

estão numeradas nas imagens - serão explicadas com mais detalhes nas subseções a seguir.

3.1 Base de dados

Para validação da hipótese levantada neste projeto, será utilizada a base de dados Middleburry 2006 (Scharstein & Pal 2007) e (Hirschmüller 2007), que contém 21 conjuntos de imagens, no qual cada conjunto dispõe de sete visões do cenário obtidas sob diferentes tipos de iluminação. A base de dados fornece os mapas de disparidade para duas dessas visões, as quais serão utilizadas como os pares de imagens estéreo. A Figura 3.3 ilustra uma amostra dessa base.

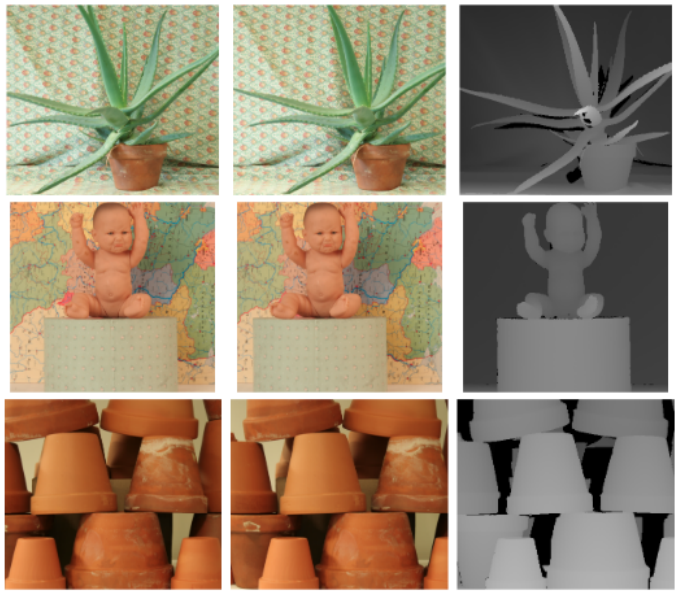


Figura 3.3: **Amostra da base de dados Middleburry 2006.** As primeira e segunda colunas representam as imagens da esquerda e direita, respectivamente. A última coluna representa o *ground-truth* do mapa de disparidade.

Os pares de imagens do banco, além de livres de distorções radiais, estão retificadas,

ou seja, elas possuem projeções epipolares correspondentes. Em outras palavras, essas imagens sofreram transformações de modo que os pontos correspondentes do cenário estão na mesma coordenada y em todas elas. Esta técnica de retificação é essencial para otimizar processos na área de Visão Computacional, uma vez que o espaço de busca pelos pontos correspondentes de um par de imagem é reduzido de duas dimensões para apenas uma dimensão. Ademais, a retificação garante que os pontos da cena possuem correspondência única nas imagens, e que todos os pontos de uma linha epipolar estão ordenados da mesma forma nas imagens retificadas, se os objetos forem opacos.

A base de dados original fornece as imagens em três resoluções. Para este trabalho, foi-se utilizada a de menor resolução, que corresponde a um terço da original. Nesta configuração, as larguras das imagens e variam entre 413 e 465 pixels, enquanto a altura de 370 é mantida para todas. A decisão de usar as imagens menores foi tomada a fim de otimizar o alto custo computacional e tempo requerido para o processamento do sistema completo.

3.2 Estimação de disparidade

O objetivo do sistema, como já citado nos capítulos anteriores, é avaliar a qualidade de mapas de disparidade, dado um algoritmo estimador. Portanto, as imagens puras que foram obtidas do banco de dados não podem ser diretamente utilizadas como entrada para a etapa de treinamento do classificador. Antes disso, elas precisam ser transformadas em mapa de disparidade. Para este processamento, será utilizado um estimador de mapa de disparidade.

Os algoritmos de estimação de mapa de disparidade têm por objetivo utilizar um par de imagens de uma mesma cena e identificar pontos pontos em ambas as imagens que correspondem ao mesmo ponto na cena.

No Capítulo 4 são sinalizados os algoritmos utilizados neste trabalho para a realização dos testes.

3.3 Processamento dos dados

Uma vez obtidos os mapas de disparidade, é preciso gerar, de fato, os exemplos de treinamento e de teste. Por se tratar de uma técnica de aprendizado supervisionado, cada exemplo deve conter um vetor de características e uma classe à qual este exemplo pertence. Neste trabalho, o conjunto de classes tem cardinalidade 2 e é composto por 0: errado, 1: correto. Portanto cada pixel de cada mapa de disparidade gerado no passo anterior foi comparado ao seu respectivo valor no *ground-truth* fornecido pela base. Cada pixel estimado corretamente foi categorizado com a classe e 1, e, do contrário, classe 0.

O critério utilizado para indicar corretude da estimação do valor da disparidade foi o mesmo utilizado em (Aristotle Spyropoulos 2014) e (Aristotle Spyropoulos 2016), isto é,

se a diferença absoluta dos valores de disparidade analisados forem maior do que 1, esta estimação é considerada incorreta. Este método é conhecida como *Left-Right Consistency* (LRC). O LRC é um indicador binário da corretude de uma correspondência da imagem da esquerda para a imagem da direita.

Para produção do vetor de características é necessário extrair um conjunto de informações relevantes o suficiente para descrever cada pixel. Em (Aristotle Spyropoulos 2014), os autores utilizaram um conjunto de oito métricas como vetor de características. Estas métricas incluem informações como a distância do pixel para a borda mais próxima da imagem, o custo mínimo das disparidades para o pixel quando obtidas as correspondências estéreo, além de medidas descritas no artigo supracitado (Hu & Mordohai 2012).

Neste trabalho, em oposição de calcular diversas métricas em relação ao pixel, serão utilizados os valores de disparidade dos pixels de uma janela cujo centro é o pixel analisado. Os experimentos apresentados nos próximos capítulos utilizam uma janela de 7x7 ao redor do pixel, o que representa um vetor de características de tamanho 49. Os 21 mapas de disparidade estimados geram um conjunto de dados de tamanho |3.243.042 x 49|. Uma vez computado o vetor de características e a classe associada à cada pixel, se é possível prosseguir para a etapa de aprendizagem.

3.4 Aprendizagem

3.4.1 Máquina de aprendizagem

Como já mencionado nos objetivos do trabalho, optou-se por utilizar máquinas de aprendizagem supervisionadas para estimar a acurácia do algoritmo de geração de mapas de disparidade. Apesar dos experimentos englobarem um conjunto de algoritmos diferentes, a estrutura dos dados é a mesma para todos os casos:

```
características:  janela 7x7 vetorizada (tamanho 49)
                 classe:  [0,1]
```

Com os dados em mãos, pode-se treinar a máquina de aprendizado sob a base de dados - ou parte dela - a fim de gerar um modelo capaz de diferenciar as classes de acordo com as características. E, a partir disso, por meio de validação cruzada, avaliar a qualidade do modelo em questão.

3.4.2 Modelo

O modelo obtido no passo anterior marca o fim da segunda etapa do sistema e o começo da terceira - e última. Sendo fornecido qualquer mapa de disparidade gerado pelo algoritmo selecionado no passo 2, o modelo deve ser capaz de identificar com alta precisão quais pixels foram estimados corretamente, ou seja, a acurácia da estimação.

3.5 Dados de testes

São os pares de imagens estéreo que serão utilizadas para avaliação do sistema. O mapa de disparidade delas será estimado usando o mesmo algoritmo da fase de treinamento e este mapa servirá de entrada para o modelo já treinado. A saída do modelo será uma classificação de certo ou errado para cada pixel do mapa de disparidade. Para aplicações reais, estas imagens não precisam ter *ground-truth*.

Capítulo 4

Experimentos

Nesta seção serão detalhadas as configurações dos experimentos realizados - e seus respectivos resultados - a fim de avaliar a performance do método proposto.

4.1 Tecnologias

O sistema foi desenvolvido por inteiro utilizando a linguagem de programação Python, desenvolvida em 1991 por Guido van Rossum (van Rossum 1995). Apesar de não ser a linguagem mais rápida para tarefas que envolvem aprendizagem de máquina, Python é bastante geral e oferece um bom tradeoff entre performance e variedade de ferramentas já implementadas.

Por ser uma linguagem multiparadigma, Python suporta conceitos de programação imperativa, funcional e orientada a objetos, além de possuir tipagem dinâmica e forte, deixando o código simples, elegante e legível. Esta foi a linguagem escolhida para o desenvolvimento do método proposto pois, além dos fatores supracitados, as bibliotecas utilizadas são oferecidas gratuitamente, a documentação é informativa e o suporte da comunidade na internet é bastante significativo.

Algumas das bibliotecas de caráter open-source foram utilizadas no projeto. Dentre elas estão:

NumPy

É um pacote da linguagem Python que suporta arrays e matrizes multidimensionais, possui um vasto leque de funções matemáticas básicas e complexas e dá suporte a operações de álgebra linear, dentre outras funcionalidades. O NumPy é frequentemente utilizado como base para construções de outras bibliotecas matemáticas em Python.

OpenCV

Nomeada a partir do termo "Open Source Computer Vision Library", que - em tradução livre - significa "Biblioteca open-source para visão computacional", é uma biblioteca multiplataforma e bastante popular entre a comunidade de visão computacional, pois possui módulos de processamento de imagem e vídeo, álgebra linear e estrutura de dados (Bradski 2000). O OpenCV, apesar de escrito em C/C++ otimizado, foi utilizado aqui com a sua interface para Python. Neste trabalho, a biblioteca em questão foi fundamental para a leitura e processamento das imagens, além dos algoritmos estimadores de mapas de disparidade.

Scikit-learn

O Scikit-learn, ou sk-learn, é o compilado de ferramentas para mineração e análise de dados mais popular entre os desenvolvedores na área de aprendizagem de máquina. Foi construída em cima do NumPy e outras bibliotecas básicas, em 2007. é possível encontrar, no sk-learn, o estado da arte de implementações de vários algoritmos populares da aprendizagem de máquina, dispondo de uma interface fácil de aplicar. Neste trabalho foram utilizados os módulos de classificação - contando com algoritmos como Naive Bayes, SVM, Random Forest, kNN, etc -, decomposição e toda a parte de avaliação das métricas - como validação cruzada e cálculo de f1.

4.2 Configuração

Todos os experimentos foram realizados utilizando o banco de dados Middleburry 2006, como citado no Capítulo 3 - que aborda a metodologia do trabalho.

Como estimadores dos mapas de disparidade, foram escolhidos três algoritmos a serem testados:

1. **SGBM:** implementação do OpenCV do algoritmo Semi Global Block Matching (Hirschmüller 2008).
2. **Local SAD:** implementação do algoritmo básico de Local Block Matching usando o SAD (*Sum of Absolute Differences*) como função de custo.
3. **Hybrid SAD:** implementação do algoritmo de Block Matching utilizando uma abordagem híbrida (baseado em bloco e em região), ainda utilizando o SAD como função de custo.

A Figura 4.1 mostra quatro exemplos dos mapas de disparidade gerados utilizando os algoritmos escolhidos.

Para a geração dos dados a partir das disparidades, foi utilizada uma janela em torno de cada pixel de dimensões $7x7$. Para definir se a disparidade deve ser rotulada com correta ou

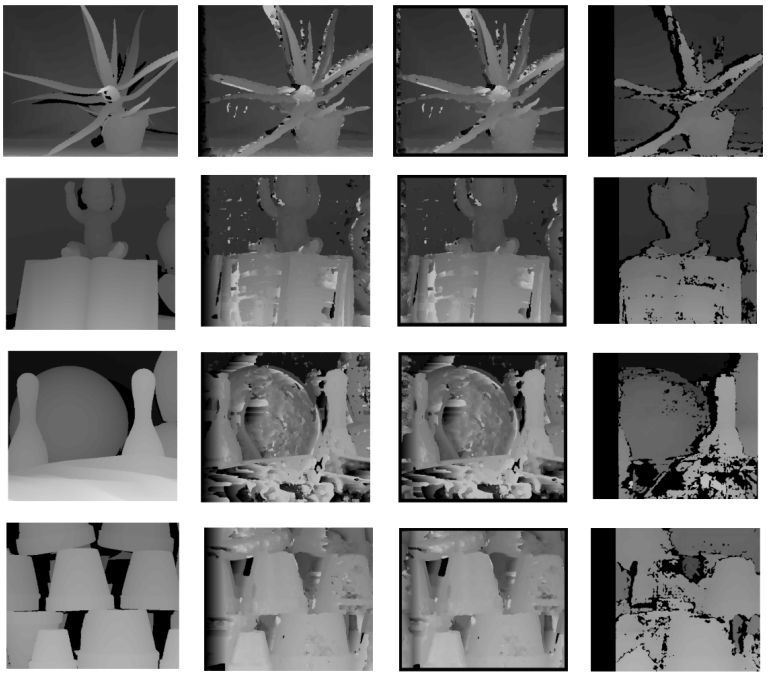


Figura 4.1: **Exemplos dos mapas de disparidade.** Na primeira coluna, são mostrados os *ground-truths* das imagens fornecidos pela base. Nas demais colunas estão os mapas de disparidade gerados pelos algoritmos Local SAD, Hybrid SAD e SGBM, respectivamente.

errada, foi medida a diferença do valor de disparidade entre o *ground-truth* e a estimação. Se essa diferença for menor que 1, a estimação é considerada correta; do contrário, é considerada errada.

Após a geração dos dados, as 21 imagens analisadas somam mais de 3 milhões de exemplos com 49 características cada. Essa quantidade de dados é bastante elevada. Portanto, para atenuar a complexidade dos dados, foi aplicada a técnica de PCA para redução de dimensionalidade. A escolha da quantidade de componentes a serem considerados foi baseada na variância que eles representam. Para os testes realizados neste trabalho, foi considerada a quantidade suficiente de componentes que representam 95% da variância total dos dados, reduzindo a dimensionalidade dos dados de 49 para 26 características.

A técnica de validação cruzada foi empregue a fim de avaliar a acurácia da etapa de classificação. As métricas foram extraídas utilizando 5 *folds*. Isto significa que o conjunto de pares de imagens foi dividido em 5 partes mutualmente exclusivas do mesmo tamanho. Daí, um subconjunto foi reservado para teste, enquanto os demais foram utilizados para a fase de treinamento. Este processo foi repetido 5 vezes. Alguns classificadores menos performáticos se tornaram bastante lentos devido à grande quantidade de dados e uma quantidade maior de *folds* não seria viável.

SGD	*Stochastic Gradient Descent*
NB	*Naive Bayes*
RF	*Random Forest*
MLP	*Multilayer Perceptron*
DT	Árvore de decisão (*Decision Tree*)
ADA	AdaBoost
QDA	*Quadratic Discriminant Analysis*
SVM	Máquina de vetores de suporte *(Support Vector Machine)*
KNN	K Vizinhos Próximos (*K Nearest Neighbors*)

Tabela 4.1: Classificadores utilizados para execução dos testes.

	Local SAD			Hybrid SAD			SGBM		
	precision	recall	f1	precision	recall	f1	precision	recall	f1
SGD	0,661	0,574	0,437	0,741	0,574	0,442	0,742	0,604	0,489
NB	0,756	0,749	0,747	0,725	0,717	0,713	0,622	0,615	0,568
RF	0,848	0,844	**0,844**	0,822	0,815	**0,815**	0,695	0,649	0,598
MLP	0,838	0,833	**0,833**	0,813	0,806	0,806	0,688	0,646	0,596
DT	0,839	0,835	**0,835**	0,813	0,806	0,806	0,680	0,645	0,595
ADA	0,839	0,834	**0,834**	0,811	0,805	0,805	0,687	0,641	0,585
QDA	0,704	0,654	0,617	0,715	0,662	0,630	0,673	0,647	0,603
SVM	0,747	0,675	0,634	0,709	0,623	0,551	0,681	0,610	0,516
KNN	0,813	0,809	0,8065	0,801	0,789	0,795	0,670	0,612	**0,639**

Tabela 4.2: Valores de precisão, cobertura e f1 para os classificadores à esquerda executados sobre os algoritmos de correspondências estéreo nas colunas. Em destaque, estão os valores mais significativos de f1 para cada algoritmo.

4.3 Resultados

Na Tabela 4.2 podem ser observados os valores resultantes da precisão, cobertura e f1 obtidos pelos classificadores à esquerda da tabela. Os classificadores utilizados se encontram nomeados na Tabela 4.1. Esses classificadores foram treinados e testados sobre mapas de disparidades oriundos de 3 algoritmos de estimação de disparidade diferentes. Os três estão identificados nas colunas triplas da tabela.

Apesar de não ter sido definido como requisito neste trabalho, o custo computacional dos métodos observados é um aspecto importante para se levar em consideração em aplicações pr·ticas. A Tabela 4.3 apresenta uma análise inicial dos tempos médios de execução para cada um destes algoritmos avaliados. O tempo de execução das fases de treinamento e teste foram coletados separadamente. Os valores apresentados foram computados em uma máquina com especificações detalhadas na Tabela 4.5.

Na Tabela 4.3 que referencia os tempos de execução, o tempo de treinamento do algoritmo KNN é diferente de zero - o que, à primeira vista, parece incorreto, visto que o KNN

	Local SAD		Hybrid SAD		SGBM	
	treinamento	teste	treinamento	teste	treinamento	teste
SGD	1,320 × T_{tr1}	1,438 × T_{te1}	2,413 × T_{tr2}	5,444 × T_{te2}	1,858 × T_{tr3}	1,054 × T_{te3}
NB	T_{tr1}	2,352 × T_{te1}	T_{tr2}	2,806 × T_{te2}	T_{tr3}	1,823 × T_{te3}
RF	63,441 × T_{tr1}	8,950 × T_{te1}	46,303 × T_{tr2}	9,195 × T_{te2}	90,514 × T_{tr3}	11,563 × T_{te3}
MLP	235,410 × T_{tr1}	8,385 × T_{te1}	192,446 × T_{tr2}	7,496 × T_{te2}	119,830 × T_{tr3}	3,619 × T_{te3}
DT	7,229 × T_{tr1}	T_{te1}	5,251 × T_{tr2}	T_{te2}	6,583 × T_{tr3}	T_{te3}
ADA	78,727 × T_{tr1}	10,233 × T_{te1}	61,678 × T_{tr2}	10,042 × T_{te2}	59,847 × T_{tr3}	10,747 × T_{te3}
QDA	2,443 × T_{tr1}	3,001 × T_{te1}	1,523 × T_{tr2}	2,598 × T_{te2}	2,843 × T_{tr3}	2,749 × T_{te3}
SVM	5886,431 × T_{tr1}	3,978 × T_{te1}	5138,635 × T_{tr2}	5,379 × T_{te2}	3849,296 × T_{tr3}	3,239 × T_{te3}
KNN	15467,829 × T_{tr1}	123642,198 × T_{te1}	14246,924 × T_{tr2}	115710,009 × T_{te2}	14441,809 × T_{tr3}	109083,200 × T_{te3}

Tabela 4.3: Tempos médios de execução, em segundos, dos classificadores à esquerda executados sobre os algoritmos de correspondências estéreo nas colunas. As variáveis representando o menor tempo são definidas na Tabela 4.4

Variável	Tempo em segundos
T_{tr1}	1,8818
T_{te1}	0,9306
T_{tr2}	2,0947
T_{te2}	0,9283
T_{tr3}	1,2607
T_{te3}	0,8813

Tabela 4.4: Variáveis utilizadas na Tabela 4.3

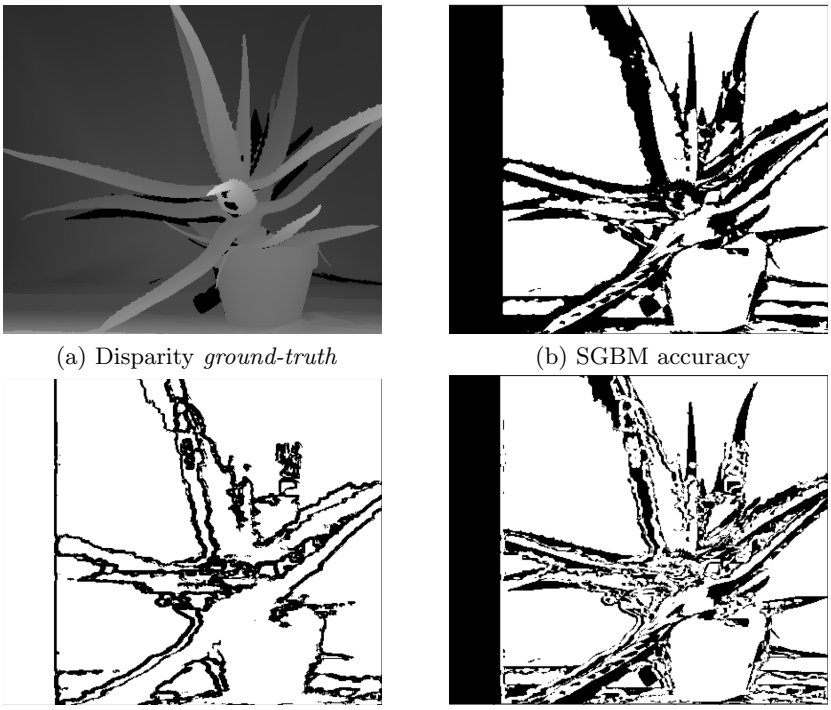
não possui fase de treinamento. Entretanto o KNN utilizado no projeto é uma implementação do Scikit, que realiza uma otimização, populando uma *k-d Tree* antes de passar para a fase de testes. Logo, a fase de treinamento exige um tempo a mais, porém isso reduz o tempo de teste.

Podemos notar, primeiramente, que os resultados dos classificadores para os dados do algoritmo SGBM foram significativamente piores do que os dos outros algoritmos. Isso se deve, principalmente a uma "barra preta" gerada pelo SGBM na margem esquerda dos mapas de disparidade, que significa que o algoritmo possui uma limitação, o impedindo de identificar disparidades deste pedaço de cena. A região mencionada pode ser observada nas imagens da última coluna da Figura 4.1.

Sistema operacional	MacOS Sierra 10.12.6
Processador	Intel Core i7
Velocidade do processador	2.5GHz
Número de cores	4
Memória	16GB

Tabela 4.5: Especificação da máquina utilizada para execução dos testes.

A Figura 4.2 mostra, em imagens, os resultados obtidos pelo algoritmo Random Forest



(a) Disparity *ground-truth* (b) SGBM accuracy

(c) Random Forest prediction (d) Prediction accuracy

Figura 4.2: Resultados do classificador Random Forest sobre o mapa de disparidade de Aloe gerado pelo algoritmo SGBM

aplicado ao mapa de disparidade gerado pelo algoritmo SGBM na imagem Aloe. A Figura 4.2(a) mostra o *ground-truth* da disparidade da imagem Aloe. Já a Figura 4.2(b) representa as labels dos pixels de entrada do classificador, ou seja, quais pixels foram corretamente (brancos) ou erroneamente (pretos) estimados pelo algoritmo SGBM. A predição do classificador Random Forest é mostrada na imagem 4.2(c) e a última imagem mostra os acertos e os erros da classificação. A faixa preta da esquerda mostra que todos esses pixels da região foram classificados erroneamente.

Por outro lado, os algoritmos Local SAD e Hybrid SAD obtiveram resultados bem semelhantes, o que é um comportamento esperado, uma vez que o objetivo deste trabalho é criar um sistema de classificação o mais genérico possível em relação aos algoritmos de estimação de disparidade.

Analisando os resultados dos classificadores, pode-se reparar os desempenhos dos classificadores Naive Bayes e KNN. Os dois classificadores, tomados como simples devido à facilidade de compreensão e implementação de ambos, apresentaram resultados competitivos com o das soluções mais complexas. O KNN, entretanto, se mostrou o algoritmo mais sensível à quantidade de dados processados, pois exigiu um tempo de execução muito superior à todos os outros testados neste trabalho.

Outro resultado um tanto surpreendente foi do método Decision Tree, que é um algoritmo também considerado simples e bastante sujeito à overfitting. Os algoritmos Random

Forest - que é a "evolução" do Decision Tree - e MLP se encontram entre os de melhores resultados. Ambos, por serem complexos e capazes de lidar com problemas não-lineares, obtiveram um resultado bom com tempo de processamento bastante baixo.

Capítulo 5

Conclusão

O principal objetivo deste trabalho foi avaliar a hipótese de que as técnicas de Aprendizagem de Máquina podem ser empregadas em áreas diversas da computação com diferentes alegações. Aqui, o uso foi motivado pela necessidade de avaliar os algoritmos de estimação de mapas de disparidade, mesmo quando as imagens não possuem groundtruth associado.

As imagens do banco de dados Middleburry, cujos pares de imagens estéreo contém o groundtruth da disparidade, foram utilizadas como entrada para os algoritmos de estimação de disparidade. Após isso, foi extraída uma janela ao redor dos pixels dos mapas obtidos e esses pixels foram rotulados, de forma a considerar a estimação correta ou incorreta. Finalmente, modelos de classificação foram treinados operando sobre os pixels obtidos, na tentativa de predizer se um pixel teve sua disparidade estimada corretamente ou não.

Os resultados mostraram que é possível obter um f1 de 84,4% sem considerar processamentos muito complexos sobre os dados ou sobre os classificadores. Esse conhecimento é relevante pois sistemas como o proposto aqui podem se tornar uma medida de melhoria de resultados para algoritmos de estimação de disparidade, de maneira independente a groundtruths.

Referências Bibliográficas

Aristotle Spyropoulos, Nikos Komodakis, P. M. (2014). Learning to detect ground control points for improving the accuracy of stereo matching, *IEEE Conference on Computer Vision and Pattern Recognition* pp. 1621–1628.

Aristotle Spyropoulos, P. M. (2016). Correctness prediction, accuracy improvement and generalization of stereo matching using supervised learning, *International Journal of Computer Vision* pp. 300–318.

Bradski, G. (2000). The OpenCV Library, *Dr. Dobb's Journal of Software Tools* .

Hartley, R. & Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*, 2 edn, Cambridge University Press, New York, NY, USA.

Hirschmüller, H. (2007). Evaluation of cost functions for stereo matching, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Hirschmüller, H. (2008). Stereo processing by semiglobal matching and mutual information, *IEEE transactions on pattern analysis and machine intelligence* **30**: 328–341.

Hu, X. & Mordohai, P. (2012). A quantitative evaluation of confidence measures for stereo vision, *IEEE transactions on pattern analysis and machine intelligence* **34**: 2121 – 2133.

Izadi, S. et al. (2011). Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera, *IEEE International Symposium on Mixed and Augmented Reality*, pp. 559–568.

Mitchell, T. M. (1997). *Machine Learning*, McGraw-Hill Education.

Scharstein, D. (1999). A survey of image-based rendering and stereo, *View Synthesis Using Stereo Vision* pp. 23–39.

Scharstein, D. & Pal, C. (2007). Learning conditional random fields for stereo.

Scharstein, D. & Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, *International Journal of Computer Vision* **47**(1-3): 7–42.

van Rossum, G. (1995). Python Tutorial, *Technical Report CS-R9526*, Centrum voor Wiskunde en Informatica (CWI).