



UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO  
UNIDADE ACADÊMICA DO CABO DE SANTO AGOSTINHO  
TECNÓLOGO EM AUTOMAÇÃO INDUSTRIAL

DAVI DE ALMEIDA MELO

Avaliação de algoritmo de clusterização para segmentação de nuvens de pontos 3D

Cabo de Santo Agostinho – PE

2022

DAVI DE ALMEIDA MELO

Avaliação de algoritmo de clusterização para segmentação de nuvens de pontos 3D

Monografia apresentada para a Unidade Acadêmica do Cabo de Santo Agostinho, campus da Universidade Federal Rural de Pernambuco para obtenção do título de Tecnólogo em Automação Industrial.

Área de concentração: Processamento Digital de Imagens

Orientador: Felipe Alberto Barbosa Simão  
Ferreira

Cabo de Santo Agostinho – PE

2022

# FICHA CATALOGRÁFICA

Dados Internacionais de Catalogação na Publicação  
Universidade Federal Rural de Pernambuco  
Sistema Integrado de Bibliotecas  
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

---

M528a      Melo, Davi de Almeida  
Avaliação de algoritmo de clusterização para segmentação de nuvens de pontos 3D / Davi de Almeida Melo. - 2022.  
47 f. : il.

Orientador: Felipe Alberto Barbosa Simao Ferreira.  
Inclui referências, apêndice(s) e anexo(s).

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Tecnólogo em Automação Industrial, Cabo de Santo Agostinho, 2022.

I. Automação. 2. Clusterização. 3. Segmentação. 4. Nuvem de Pontos. I. Ferreira, Felipe Alberto Barbosa Simao, orient.  
II. Título

CDD 629.895

---

DAVI DE ALMEIDA MELO

Avaliação de algoritmo de clusterização para segmentação de nuvens de pontos 3D

Monografia apresentada a Unidade Acadêmica do Cabo de Santo Agostinho da Universidade Rural de Pernambuco para obtenção do título de Tecnólogo em automação industrial

Aprovada em: 05/10/2022

Banca Examinadora

---

Orientador

---

Examinador Interno

---

Examinador Interno

## RESUMO

As tecnologias 3D têm sido exploradas nas diversas áreas da indústria. Com essas tecnologias são executadas funções como visualização, instrumentação, controle, simulação, treinamento, planejamento, documentação, entre outros. A partir disso, novos tipos de mídia foram introduzidos a esse contexto industrial. Como exemplo, existem as nuvens de pontos, que trata-se de um conjunto de pontos distribuídos num modelo tridimensional da realidade. Elas, geralmente, são construídas através da atuação de um *scanner* e podem conter em cada um de seus pontos as características de um objeto como localização, cor, reflectância e entre outros aspectos. Dado que as nuvens de pontos irão representar peças, equipamentos, tubulações, máquinas, áreas, e estruturas no contexto industrial, conseguir segmentar as nuvens de pontos e possibilitar uma melhor visualização das partes separadas da mesma estrutura é uma ferramenta proveitosa. Além disso, dentre as funções das tecnologias 3D apresentadas, a segmentação de nuvens de pontos perpassa de forma direta e indireta as áreas de visualização, instrumentação e controle. Portanto, verificada a importância da segmentação de nuvens de pontos, o objetivo desta monografia é avaliar dois algoritmos de clusterização para segmentação de nuvens de pontos 3D. São eles, o DBSCAN e o *K-means*. Ambos estão categorizados como algoritmos de clusterização por meio aprendizagem de máquina não supervisionada. Após a avaliação, foram constatadas as diferenças entre cada algoritmo. Verificou-se um melhor desempenho por parte do *K-means* quando se trata de dados dispersos e o equivalente para o DBSCAN quando se referem a distribuição de dados com certa distância entre os *clusters*.

Palavras-chave: automação; clusterização; segmentação; nuvem de pontos.

## **ABSTRACT**

3D technologies have been explored in many industrial fields. With those technologies functions such as visualization, instrumentation, control, simulation, training, planning, documentation, and more are performed. From that, new types of media were introduced to the industrial context. For instance, the point clouds, which is a set of points distributed on a tridimensional model of reality. They are often built through the operation of a scanner and may contain the characteristics of an object, such as localization, color, reflectance and more in every one of its points. Given that the point clouds represent pieces, equipments, piping, machines, areas and structures in the industrial context, a useful tool is to segment the point clouds and enable a better visualization of separate parts of the same structure. Furthermore, among the presented 3D technologies, point cloud segmentation permeates directly and indirectly the fields of visualization, instrumentation, and control. Thus, verified the importance of the point cloud segmentation, the goal of this monography is to assess two clustering algorithms for 3D point cloud segmentation. They are the DBSCAN and the K-means. Both are categorized as clustering algorithms through unsupervised machine learning. After the assessment, the differences between each algorithm were found. The K-means was verified to have better performance when it comes to disperse data and the equivalent to the DBSCAN when it comes to data distribution with a certain distance between the clusters.

**Keywords:** automation; clustering; segmentation; point clouds.

## LISTA DE FIGURAS

<b>Figura 1</b> - Nuvem de pontos industrial .....	12
<b>Figura 2</b> - Implementação do <i>K-means</i> em <i>Python</i> . .....	28
<b>Figura 3</b> - Nuvem de pontos Bumbameuboi. ....	29
<b>Figura 4</b> - Nuvem de pontos Bumbameuboi segmentada em dois <i>clusters</i> .....	29
<b>Figura 5</b> – Nuvem de Pontos Guy .....	31
<b>Figura 6</b> – Nuvem de Pontos Quarto .....	31
<b>Figura 7</b> – Distribuição de dados A.....	35
<b>Figura 8</b> – Distribuição de dados B .....	35
<b>Figura 9</b> – Nuvem de Pontos Quarto Segmentada pelo DBSCAN .....	36
<b>Figura 10</b> – Nuvem de pontos Bumbameuboi Segmentada pelo DBSCAN .....	37
<b>Figura 11</b> - Nuvem de pontos Guy Segmentada pelo DBSCAN.....	37
<b>Figura 12</b> – Nuvem de pontos Quarto segmentada pelo <i>K-means</i> .....	38
<b>Figura 13</b> – Nuvem de pontos Bumbameuboi segmentada pelo <i>K-means</i> .....	38
<b>Figura 14</b> – Nuvem de pontos Guy segmentada pelo <i>K-means</i> .....	39

## SUMÁRIO

<b>1 INTRODUÇÃO</b>	10
<b>2 OBJETIVOS</b>	11
2.1 OBJETIVO GERAL	11
<b>3 FUNDAMENTAÇÃO TEÓRICA</b>	12
3.1 POINT CLOUD	12
3.2 SEGMENTAÇÃO	13
3.3 ALGORITMOS	14
3.3.1 K-means	14
<b>3.3.1.1 Introdução</b>	14
<b>3.3.1.2 Inicialização</b>	15
<b>3.3.1.3 Particionamento</b>	15
<b>3.3.1.4 Teste de parada</b>	16
<b>3.3.1.5 Atualização</b>	16
3.3.2 DBSCAN	17
3.4 MÉTRICAS DE AVALIAÇÃO	20
3.4.1 Homogeneidade	20
3.4.2 Completude	21
3.4.3 V-measure	22
3.4.4 Acurácia	23
3.5 DISTÂNCIAS	23
3.5.1 Distância Euclidiana	23
3.5.2 Distância Euclidiana Quadrática	23
3.5.3 Distância <i>City Block</i> (distância de Manhattan ou distância absoluta)	24
3.5.4 Distância de Bray Curtis (ou distância Sorrensen)	24
3.5.5 Distância de Canberra	24
<b>4 METODOLOGIA</b>	24
4.1 BANCO DE DADOS	24
4.1.1 Adaptação de banco de dados para segmentação não semântica	25
4.1.2 Nuvem de pontos não industriais	25
4.1.3 Decisões e possíveis pesquisas	26
4.2 IMPLEMENTAÇÃO DO <i>K-MEANS</i>	27
4.3 UTILIZAÇÃO DO DBSCAN	30



4.4 TESTES E COMPARAÇÕES	30
4.4.1 Teste 1: Comparação de algoritmos com nuvem de pontos 2D	30
4.4.2 Teste 2: Comparação visual de algoritmos com nuvem de pontos 3D	30
4.4.3 Teste 3: Comparação visual entre tipos de distância no <i>K-means</i>	31
<b>5 RESULTADOS E DISCUSSÃO</b>	31
5.1 K-MEANS E DBSCAN	31
5.1.1 Teste 1	31
5.1.2 Teste 2	36
5.1.3 Teste 3	39
<b>6 CONCLUSÃO</b>	41
<b>REFERÊNCIAS</b>	42

## 1 INTRODUÇÃO

Como discutido em Fonseca (2013), as tecnologias 3D possuem um vasto campo de atuação dentro do contexto da automação industrial. De forma análoga, está o uso da inteligência artificial em conjunto com a automação que possibilita melhores resultados e mais produtividade em cada processo da indústria 4.0. Um bom exemplo disso é o potencial acréscimo do número de sensores em um equipamento devido as aplicações com IA (Inteligência Artificial) como apresentado em Zheng (2018 apud RIBEIRO, 2020, p. 53). Além de auxiliar nos fatores de crescimento e produtividade, as tecnologias imersivas podem colaborar com um dos procedimentos frequentes desse ambiente, que é a manutenção e prevenção de segurança dos equipamentos. Para tal, os DTs (*digital twins* ou irmãos digitais) são um conjunto de mídias que representam estruturas e objetos da realidade, eles podem ser manipulados digitalmente em *softwares* como *Autodesk*, *AVEVA* e *Bently* (AGAPAKI, 2021). Essas manipulações ocorrem, por exemplo, no intuito de segmentar os DTs para melhor visualização, planejamento, manutenção, entre outros. Contudo, de acordo com Agapaki (2021), os irmãos digitais não são utilizados com facilidade, pois, apesar de proporcionar benefícios, o seu custo de geração e manutenção não equivale as vantagens. Desse modo, fica claro a necessidade de pesquisa sobre o conjunto de novas mídias geradas nesse contexto e a capacidade para serem utilizadas e manipuladas pela área da aprendizagem de máquina.

A segmentação de uma PC (*Point Cloud*) ou nuvem de pontos, é uma operação que pode ser constituída por um algoritmo de clusterização supervisionado ou não (XIE, 2020). Uma PC faz parte das mídias que podem se tornar irmão digital de um objeto. Ela é um conjunto de pontos dispostos em um espaço tridimensional que busca representar a realidade e trata-se, atualmente, de um dos principais tipos de mídia das tecnologias imersivas (XU, 2018). Essa mídia está em ascensão não só no contexto industrial, mas em diversas áreas como pode ser constatado em Jung (2020). Uma nuvem de pontos pode representar máquinas, tubulações, áreas, peças e estruturas no geral. Já a segmentação é uma operação que envolve o agrupamento de dados, ela pode ter como critério diversos parâmetros da PC como distância entre pontos, cor dos pontos, densidade, reflectância etc. Além disso, uma segmentação pode ser com base em um procedimento semântico ou não. Ou seja, é possível dividir os tipos de segmentação de nuvem de pontos em dois grupos, PCSS (*Point Cloud Semantic Segmentation*) e PCS (*Point Cloud Segmentation*) (XIE, 2020).

A segmentação semântica divide os agrupamentos no que será denominado por *labels*, classes, ou ainda, rótulos. Cada *label* possui um valor em texto a qual o identifica. Como exemplo, dado que temos certa nuvem de pontos que caracteriza uma mesa com vaso de flores e dois pratos, as *labels* seriam as seguintes: mesa, vaso, flor e prato (MEIRA, 2020). Já a segmentação não semântica cumpre o mesmo processo de identificar objetos e dividir a PC de acordo com eles, no entanto, não atribui classes com um contexto semântico aos agrupamentos; cada ponto é agrupado com rótulos numerados. Uma PCSS precisa utilizar uma técnica de classificação supervisionada, enquanto uma PCS pode ser resolvida a partir de técnicas de clusterização não supervisionadas. Cada uma dessas condições corrobora nas vantagens e desvantagens de cada técnica. A PCS, geralmente, sendo o processo mais simples e menos computacionalmente custoso, enquanto a PCSS pode alcançar maior precisão devido ao complexo aprendizado de máquina que atravessa. Neste trabalho, o estudo será voltado a segmentação não semântica, PCS. Esta, será desenvolvida a partir dos algoritmos de clusterização *K-means* e DBSCAN. O tipo de algoritmo denominado por clusterização está relacionado a constituição de agrupamentos que possuam alta similaridade entre os pontos de um mesmo grupo e ou baixa similaridade entre pontos de grupos diferentes.

Outros trabalhos já exploraram a segmentação para o contexto industrial como abordado em Agapaki (2021), Yin (2021) e Zermas (2017). Além disso, existe uma grande utilização do algoritmo *K-means* na indústria e em contexto análogo facilmente adaptável como demonstrados nos trabalhos de Sampath (2006), Bargoni (2022) e Ying (2022). Contudo, não foram encontrados trabalhos que avaliassem a segmentação não semântica utilizando algoritmos de aprendizagem de máquina não supervisionada como *K-means* e DBSCAN em comparação. É neste contexto em que está inserido o presente trabalho.

## **2 OBJETIVOS**

### **2.1 OBJETIVO GERAL**

Analisar técnicas de segmentação de nuvens de pontos 3D baseadas em algoritmos de clusterização, mais especificamente os algoritmos *K-means* e DBSCAN, para aplicação no contexto de automação industrial.

### **2.2 OBJETIVOS ESPECÍFICOS**

- Estudar e analisar técnicas de segmentação para nuvens de pontos 3D;

- Implementar o processo de segmentação de nuvem de pontos *3D* com o algoritmo de clusterização *K-means* e DBSCAN;
- Avaliar diferentes tipos de distâncias no processo de segmentação;
- Avaliar e realizar uma comparação dos resultados de segmentação.

### 3 FUNDAMENTAÇÃO TEÓRICA

#### 3.1 POINT CLOUD

Uma nuvem de pontos *3D* é um conjunto de pontos em um sistema tridimensional definido geralmente pelas coordenadas *X*, *Y* e *Z*. Cada coordenada de uma nuvem de pontos pode conter várias informações como cor, reflectância e outras características possíveis para um elemento pontual. Desse modo, uma nuvem de pontos busca caracterizar a superfície de algum objeto, por exemplo. Essa, pode ser obtida a partir de várias técnicas, por exemplo, o *scanner Lidar (Light Detection and Ranging)* que é um método que utiliza o laser do *scanner* para capturar as características de um objeto. Pulsos combinados de diferentes distâncias constroem a forma do espaço estudado. Desse modo, na indústria, as nuvens de pontos podem representar máquinas elétricas, peças mecânicas, tubulações, áreas, e estruturas no geral. Esse aspecto permite a documentação de objetos em um nível digital. O que pode facilitar a comunicação entre setores, planejamento, e outros aspectos. Logo, com o avanço nas técnicas para obtenção da PC e o acréscimo das ferramentas para lidar com esse tipo de dado, a *Point Cloud* passa a ser uma das partes mais importantes das tecnologias imersivas (XU, 2018; WHAT..., 2021; Wang, 2019). A Figura 1 retirada de Yusuf (2020), exemplifica uma nuvem de pontos no contexto industrial.

Figura 1 - Nuvem de pontos industrial



Fonte: Nuvem de pontos retirada de Yusuf (2020).

### 3.2 SEGMENTAÇÃO

O processo de segmentação é um passo importante dentro do espectro de controle de sistemas. Como exemplo, a ZIVID (ABOUT...,202-?), é uma empresa que trabalha com braços mecânicos autônomos para cumprimento de tarefas como a montagem de equipamentos, coleta de objetos, manutenção de máquinas específicas etc. Este equipamento robótico precisa obter a informação da posição dos objetos. Para caracterização da realidade a ZIVID utiliza um *scanner* para produzir uma nuvem de pontos. Logo, a segmentação da nuvem de pontos proporciona a capacidade ao braço robótico de distinguir os objetos em determinado local. Além disso, mais aplicações de como a segmentação de nuvem de pontos pode ser explorada no contexto de automação e controle de sistemas pode ser verificado em Su (2016).

Conforme mencionado anteriormente, a segmentação de nuvem de pontos pode ser dividida em duas grandes áreas, PCS e PCSS. Por sua vez a segmentação de nuvem de pontos não semântica (PCS) pode ser particionada em 4 tipos: *edge based*, *region growing*, *model fitting* e *clustering based*. A segmentação *edge based*, ou baseado em borda, aproveita da situação em que a nuvem de pontos possui bordas que definem a estrutura. Desse modo a segmentação é solucionada identificando as bordas e os pontos próximos a ela. *Region Growing*, ou crescimento de região, identifica a segmentação a qual supõe ou assume que naquela nuvem os pontos próximos possuem características parecidas. Dada esta colocação e sabendo que um *cluster* trata-se de um agrupamento de pontos da nuvem, pontos são escolhidos de forma aleatória ou não como *seeds* (sementes). As sementes serão utilizadas para agrupar todos os pontos ao seu redor que possuem grande similaridade com elas. A similaridade é nessa etapa definida, quantitativamente, sobre o quão parecido é determinado parâmetro de um ponto a outro. Ou seja, envolve a distância entre os pontos ou algum outro aspecto da nuvem para que exista uma real comparação. O *model fitting*, ou encaixe de modelo (traduzido livremente), trabalha segmentando com ênfase nas formas geométricas. Portanto, dado que temos as formas geométricas definidas, os *clusters* são criados sempre que esses modelos são detectados na estrutura. *Unsupervised Clustering Based Segmentation*, ou segmentação não supervisionada baseada em agrupamento, é um método em que os algoritmos não estão alinhados numa única lógica matemática. Trata-se de uma mistura do conjunto de métodos vistos anteriormente. Além disso, denomina-se método não supervisionado aquele que recebe pouca ou quase nenhuma informação sobre o conjunto de dados que irá tratar. Entre os principais algoritmos desse tipo de

segmentação temos o *K-means*, *mean-shift*, *fuzzy clustering* e DBSCAN (CERUTTI, 2013; XIE, 2020).

### 3.3 ALGORITMOS

O *K-means* é organizado com uma quantidade pré-determinada  $K$  de agrupamentos. Os representantes podem ser inicialmente escolhidos aleatoriamente. Um grupo que possui o representante  $X$  é construído pelos pontos mais próximos a ele. A métrica de proximidade pode ser, por exemplo, a distância euclidiana. Após a formação dos agrupamentos, novos centros são escolhidos a partir da média aritmética dos pontos de cada agrupamento. Em seguida, a etapa de associação de pontos ocorre novamente. O algoritmo segue até que os parâmetros de parada sejam atendidos (XIE, 2020; KUMAR, 2020; FONSECA, 2018). Já o DBSCAN (*Density-based spatial clustering of applications with noise*) ou traduzido livremente, clusterização espacial baseada em densidade para aplicações que possuem ruído, é um algoritmo de clusterização focado na densidade dos agrupamentos e na possível ocorrência de ruídos em meio a esses dados.

#### 3.3.1 K-Means

##### 3.3.1.1 Introdução

Abordado brevemente em Xie (2020), para nuvem de pontos, e detalhado por Fonseca (2018), no contexto de quantização vetorial, o algoritmo *K-means* pode ser dividido em 4 etapas bem definidas. Para iniciar os passos desse modelo algorítmico é preciso escolher um valor numérico inteiro  $K$  que irá definir o número de grupos que o *K-means* irá construir. O objetivo desse modelo é descobrir os  $K$  melhores agrupamentos possíveis em uma determinada distribuição de dados. O conceito de melhores agrupamentos pode ser definido em dois critérios simples. Primeiro, os elementos de um mesmo grupo devem possuir a maior similaridade possível entre si. Segundo, os elementos de grupos diferentes devem possuir baixa similaridade. Caso os agrupamentos formados cumpram essas condições, então uma boa clusterização foi conduzida e o *K-means* cumpriu seu objetivo. Para avaliar a ação do modelo de forma quantitativa, conceitos como completude, acurácia e homogeneidade serão introduzidos mais adiante. Além disso, no intuito de facilitar o entendimento prático do *K-means* aplicado em nuvem de pontos, todas as suas etapas irão ser direcionadas a essa mídia. Os pontos de uma nuvem são organizados de modo a compor vetores o conjunto desses

vetores é denominado conjunto de treino ou vetores de treino. Um vetor da nuvem pode ser representado da seguinte maneira  $(\alpha, \beta, \gamma, \delta)$ , onde  $\alpha$ ,  $\beta$  e  $\gamma$  são as coordenadas do espaço tridimensional e  $\delta$  é mais um valor numérico que pode ser qualquer informação obtida sobre aquele ponto.

Os  $K$  agrupamentos são construídos a partir de  $K$  vetores iniciais, individualmente, cada um desses vetores é chamado de representante. Analogamente, encontrar os  $K$  melhores representantes, significa o mesmo que encontrar os  $K$  melhores agrupamentos. A busca pelos melhores representantes segue em função da regra do VMP (vizinho mais próximo), que funciona encontrando o grupo dos vetores de treino mais próximos do  $j$ -ésimo vetor dos representantes. A proximidade dos vetores é calculada utilizando a distância euclidiana quadrática. A média do grupo de vetores de treino mais próximos do  $j$ -ésimo vetor dos representantes é denominado centróide. Devido a isso, eles também são chamados de centros de cada agrupamento. A região composta pelo grupo de vetores de treino mais próximos de um dado centróide, chama-se região de Voronoi. Este centróide, irá compor o novo grupo de representantes.

### 3.3.1.2 Inicialização

Trata-se da escolha de  $K$  representantes iniciais para o modelo. Neste primeiro momento, pode ser feito de forma totalmente aleatória, contudo, um modo mais otimizado em relação ao anterior é a captura aleatória de pontos da nuvem como representantes. Esta etapa é importante pois o *K-means* é altamente sensível a escolha inicial dos  $K$  vetores.

### 3.3.1.3 Particionamento

Esta etapa inclui a associação dos vetores da nuvem aos melhores representantes a partir da regra do vizinho mais próximo (VMP), que para esse caso foi construído a partir da distância euclidiana quadrática. A busca do VMP inicia ao calcular-se a distância euclidiana de cada vetor escolhido na inicialização para cada vetor de treino, encontrando dessa forma a relação de proximidade entre os vetores, que também pode ser chamada de peso. Dado que a lista com os vetores de treino é nomeada de *training set*, foi definido um vetor chamado *partition*, que armazena qual o índice do vetor entre os vetores representantes é mais próximo do  $n$ -ésimo vetor de treino. Logo, o vetor *partition* possui o mesmo tamanho que o vetor *training set*. Ou seja, o primeiro elemento do vetor *partition*, por exemplo, com índice 0, possuirá armazenado o índice

do vetor representante mais próximo ao vetor de treino de índice 0. É importante destacar que esta é a etapa mais computacionalmente custosa do algoritmo; isso ocorre devido a quantidade elevada de multiplicações e adições que são realizadas no cálculo do VMP com o uso da distância euclidiana.

#### **3.3.1.4 Teste de parada**

Trata-se de uma verificação para finalizar o algoritmo, onde definimos quão próximos os representantes estão no que se refere à similaridade com os vetores da nuvem de pontos. Para avaliar essa questão tem-se um valor denominado distorção geral. A distorção geral é a soma do módulo das diferenças entre os vetores da nuvem e os vetores representantes. Cada ciclo do algoritmo gera uma distorção geral, a partir do momento que a diferença entre as duas últimas distorções gerais for menor que um valor pré-estabelecido no algoritmo, o procedimento se encerra.

#### **3.3.1.5 Atualização**

Enquanto o teste de parada não for satisfeito, esta é a etapa responsável pela modificação e aprimoramento dos centros, onde buscaremos o centróide de cada um dos agrupamentos e tomaremos eles como os novos representantes, descartando os anteriores. Os centróides são a média dos vetores de um agrupamento formado, eles são calculados a partir do vetor *partition*. Por exemplo, seleciona-se todos os vetores de treino relacionados ao vetor de índice 0 dos vetores representantes. Calcula-se a média dos vetores de treino selecionados e esta média será o novo vetor representante com índice 0. Assim segue o algoritmo até que todos os vetores representantes tenham sido atualizados. Como os vetores representantes foram modificados, novos relacionamentos de proximidades serão necessários, logo, a etapa de particionamento será executada novamente. A partir disso, o processo segue em ciclo até que se cumpra o exigido no teste de parada. Com exceção da inicialização, pois, a etapa atual já determina os novos elementos do conjunto de vetores representantes.

Em seguida é apresentado o pseudocódigo do algoritmo *K-means* com os passos descritos anteriormente:



---

**Algoritmo *K-means* Pseudocódigo**

---

Geração de  $N$  centróides (representantes dos agrupamentos)

**ENQUANTO** verdadeiro **FAÇA:**

    para cada vetor de treino, encontrar o representante mais próximo

    cálculo da distorção geral

**SE** condição de parada for satisfeita **ENTÃO:**

        fim do algoritmo

**FIM DO SE**

    atualização dos representantes

**FIM DO ENQUANTO.**

---

**Modificado de (Fonseca, 2018)**

### 3.3.2 DBSCAN

Modelo de algoritmo baseado em *cluster* não supervisionado, o DBSCAN possui como característica principal a observação e utilização da informação de densidade do conjunto de pontos. Previamente, já abordamos o fato de ser um algoritmo não supervisionado. Em outras palavras isso significa que o DBSCAN não recebe como parâmetro quantos agrupamentos devem ser feitos, nem recebe um treinamento semântico, por exemplo, para definir em aspectos qualitativos de como são os grupos que serão colocados a sua lógica. Citando um caso contrário, digamos que seja preciso fazer a segmentação de imagens e que é necessário identificar os objetos dela. Na segmentação semântica supervisionada, o modelo irá receber um banco de dados com imagens na qual a segmentação foi feita anteriormente por um especialista. Ou seja, recebe o conjunto de dados e a resposta correta para a segmentação. O algoritmo treinado com essa distribuição de dados e respostas está preparado para segmentar um objeto específico que não faz parte do conjunto de treinamento. No DBSCAN isso não ocorre, nem são utilizadas informações conhecidas do conjunto de dados, como por exemplo, o número de *clusters* (ESTER, 1996; XIE, 2020).

O objetivo do DBSCAN é encontrar os melhores  $K$  agrupamentos baseados na informação de densidade da nuvem de pontos. Para esse propósito é preciso definir dois parâmetros iniciais, o *epsilon*, raio de uma esfera centrada em um ponto da nuvem, e *min points* que é um parâmetro de classificação. A lógica do modelo pode ser definida pelos

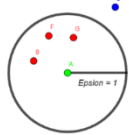
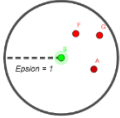
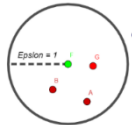
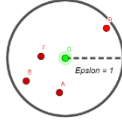
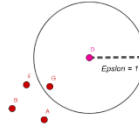

dois parâmetros atuando de forma simultânea. Este algoritmo não foi implementado pelo autor, mas está disponível em várias bibliotecas *Python* como *Scikit-Learn* e *Open3D*. Uma possível forma simplista de implementar será abordado a seguir assim como a explicação sobre o funcionamento prático do algoritmo.

Primeiro, um ponto que ainda não participa de um agrupamento e nunca foi verificado pelo algoritmo é escolhido de forma aleatória, ele será denominado ponto inicial ou ponto de verificação. Isso pode ser feito criando um vetor marcador com o mesmo tamanho do número de pontos da nuvem de pontos. Assim que um ponto for verificado ele recebe o valor 1 na sua posição. Caso contrário irá possuir o valor 0.

Após a escolha do ponto inicial, é verificado se existem pontos a uma distância *epsilon* do ponto inicial. A partir disso existem 3 possíveis situações, o ponto verificado pode ser classificado como um ponto de borda, um ruído, ou um ponto de agrupamento. Se existir um número de pontos ao redor da esfera de raio *epsilon* menor que o número inteiro definido em *min points*, o ponto verificado é classificado como ponto de borda. Se o número de pontos for maior ou igual a *min points*, será classificado como ponto de agrupamento. Por sua vez, se não forem encontrados pontos dentro da esfera, o DBSCAN categoriza o ponto como ruído.

Dado que a classificação foi finalizada para o ponto inicial, o algoritmo passa a verificar os pontos ao redor do ponto verificado. Dessa forma os pontos de borda delimitam o fim do agrupamento. A partir desse procedimento, assim que todos os pontos são verificados e agrupados o algoritmo se encerra para o primeiro agrupamento. Caso ainda exista pontos que nunca foram verificados o algoritmo poderá consultar pelo vetor marcador e a partir disso escolher aleatoriamente um deles e recomeçar a lógica descrita acima. Visando esclarecer o processo de classificação feito pelo DBSCAN o Quadro 1 foi elaborado.

Quadro 1 – Classificação com DBSCAN

Ponto Classificado	Classificação
A	
B	
F	
G	
D	
E	

Fonte: O Autor (2022)

Dado que temos essa nuvem de configuração 2D, a esfera será demarcada pela intersecção dela com o plano da nuvem de pontos. A partir disso, foi definido que neste processo de classificação o parâmetro *min\_points* será igual a 3. Ou seja, se existirem no mínimo 3 pontos dentro da esfera de classificação, o ponto avaliado será um ponto de agrupamento. Na classificação dos pontos A, B, F e G, é exatamente isso que ocorre. Detalhando o processo para o ponto A, temos que: primeiro o ponto A foi escolhido, em seguida a esfera de raio *epsilon* foi traçada, por terceira ação, contabilizou-se o número de pontos dentro da esfera, por fim a classificação foi feita a partir do parâmetro *min\_points* determinado antes do processo iniciar. O ponto D possui menos do que o mínimo de pontos necessário dentro de sua esfera para ser classificado como ponto de agrupamento, logo é um ponto de borda. Já o ponto E não possui pontos dentro de sua esfera, portanto, é classificado como ruído.

É possível compreender a nomenclatura do algoritmo com mais propriedade agora, DBSCAN (*Density-based spatial clustering of applications with noise*), ou seja, um algoritmo baseado em densidade para clusterização que envolva ruído. Por fim, a abordagem e sugestão acima foi baseada em Ester (1996) e Chen (2019) no qual o tema é aprofundado.

### 3.4 MÉTRICAS DE AVALIAÇÃO

Avaliar os algoritmos de forma quantitativa é um fator de grande importância nessa pesquisa. Para tal, esta seção dedica-se a explicar os conceitos de comparação que serão empregados mais adiante e foram catalogados a partir de Rosenberg (2007). Para âmbito de explicações posteriores, o termo classe é definido nesta seção como o título ou valor que representa o agrupamento correto de uma dada amostra. Como exemplo, na segmentação de uma imagem, teremos a informação de cada pixel e a qual classe ele pertence antes que essa, por sua vez, passe pelo algoritmo de segmentação para fins de validação. Objetos de mesma classe devem estar agrupados no mesmo *cluster* (conjunto ou agrupamento).

#### 3.4.1 Homogeneidade

A homogeneidade ideal ocorre quando todos os pontos de um *cluster* possuem a mesma classe. Quando todos os *clusters* formados na segmentação obedecem a essa lógica, a homogeneidade possui o valor máximo. Dado que *C* é a *label* de uma amostra

do conjunto de dados e  $K$  é o *cluster* dessa amostra, a equação para a homogeneidade é dada por

$$h = \begin{cases} 1 & \text{se } H(C, K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & \text{caso contrário,} \end{cases} \quad (1)$$

sabendo que

$$H(C|K) = - \sum_{k=1}^{|K|} \sum_{c=1}^{|C|} \frac{a_{ck}}{N} \log \frac{a_{ck}}{\sum_{c=1}^{|C|} a_{ck}} \quad (2)$$

e

$$H(C) = - \sum_{c=1}^{|C|} \frac{\sum_{k=1}^{|K|} a_{ck}}{n} \log \frac{\sum_{k=1}^{|K|} a_{ck}}{n}, \quad (3)$$

em que:

- $C = \{c_i | i = 1, \dots, n\}$  é o conjunto das classes de uma distribuição de dados;
- $K = \{K_i | i = 1, \dots, m\}$  é o conjunto de grupos de uma distribuição de dados.

Ou seja, na métrica de homogeneidade quanto mais próximo de 1 mais homogênea foi a segmentação feita.

### 3.4.2 Completude

A completude é um conceito simétrico a homogeneidade. Quando todos os pontos de uma dada classe possuem o mesmo *cluster* a completude é máxima. A fórmula da completude é dada por:

$$h = \begin{cases} 1 & \text{se } H(K, C) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & \text{caso contrário} \end{cases} \quad (4)$$

Sabendo que:

$$H(K|C) = - \sum_{c=1}^{|C|} \sum_{k=1}^{|K|} \frac{a_{ck}}{N} \log \frac{a_{ck}}{\sum_{k=1}^{|K|} a_{ck}} \quad (5)$$

$$H(C) = - \sum_{k=1}^{|K|} \frac{\sum_{c=1}^{|C|} a_{ck}}{n} \log \frac{\sum_{c=1}^{|C|} a_{ck}}{n} \quad (6)$$

em que:

- $C = \{c_i | i = 1, \dots, n\}$  é o conjunto das classes ou *labels* de uma distribuição de dados;
- $K = \{K_i | i = 1, \dots, m\}$  é o conjunto de grupos de uma distribuição de dados.

O valor máximo da completude é 1, quanto mais próximo desse valor melhor é a completude da distribuição de dados.

### 3.4.3 V-measure

Trata-se da média harmônica entre homogeneidade e completude. A fórmula para *V-measure* é dada por

$$V_{\beta} = \frac{(1 + \beta) \times h \times c}{(\beta \times h) + c}, \quad (7)$$

em que o parâmetro  $\beta$  é um número que pertence aos reais,  $h$  é a homogeneidade e  $c$  é a completude. Quando  $\beta$  é maior do que 1 a completude é mais importante na equação. Caso  $\beta$  seja menor do que 1, a homogeneidade tem um peso maior na equação. Para o caso em que  $\beta$  é igual a 1, nenhuma das métricas possui maior peso e a equação fica da seguinte forma.

$$V = 2 \frac{h \times c}{h + c}. \quad (8)$$

Observe que a *v-measure* é uma métrica que independe de número de *clusters*, classes e tamanho da distribuição de dados. Logo, esses parâmetros podem ser utilizados para qualquer conjunto de dados.

#### 3.4.4 Acurácia

Sabendo a qual agrupamento ou *cluster* cada ponto deveria ser classificado, ou seja, a resposta ideal da segmentação, a medida de acurácia irá trazer o percentual de acerto da segmentação feita pelo algoritmo. Trata-se de um valor que no caso ideal é igual a 1. Como exemplo, se 100 pontos são segmentados e a acurácia aponta 0,95, significa que 95 pontos foram agrupados corretamente.

### 3.5 DISTÂNCIAS

Como apresentado anteriormente, os algoritmos constroem sua relação de proximidade durante a busca do vizinho mais próximo a partir da distância euclidiana, por exemplo. Parte do desempenho obtido pelo algoritmo está relacionado a escolha do tipo de distância. Esse tipo de análise foi abordado profundamente por Suwilo (2019) e Faisal (2019). Além disso, as informações sobre distâncias a seguir também estão baseadas em Banda (2011) e Moraes (2016), pois, nesses materiais há uma breve passagem por diversos tipos de distância.

#### 3.5.1 Distância Euclidiana

Trata-se da distância entre dois pontos a partir de uma linha reta. Essa, por conseguinte, é uma das formas de avaliar a distância no espaço euclidiano. A fórmula para a distância euclidiana é dada por

$$d_{ij} = \sqrt{\sum_{k=1}^n (x_{ik} - y_{jk})^2}. \quad (9)$$

em que qual  $d$  é a distância entre duas observações  $i$  e  $j$ . Essa distância corresponde à raiz quadrada da soma dos quadrados das diferenças entre os pares de observação para todas as  $n$  variáveis. Em que  $x_{ik}$  é o valor da variável  $k$  referente à observação  $i$  e  $y_{jk}$  para a observação  $j$ .

#### 3.5.2 Distância Euclidiana Quadrática

A distância entre duas observações  $i$  e  $j$ . Esta é a soma dos quadrados das diferenças entre os pares de observações para todas as  $n$  variáveis.

$$(d_{ij})^2 = \sum_{k=1}^n (x_{ik} - y_{jk})^2 \quad (10)$$

### 3.5.3 Distância *City Block* (distância de Manhattan ou distância absoluta)

Trata-se da soma das diferenças absolutas entre duas coordenadas. A fórmula é dada por:

$$d_{ij} = \sum_{k=1}^n |x_{ik} - y_{jk}| \quad (11)$$

### 3.5.4 Distância de Bray Curtis (ou distância Sorrensen)

É uma modificação da distância de Manhattan. Ocorre quando a soma das diferenças absolutas é normalizada pela soma das coordenadas.

$$d_{ij} = \sum_{k=1}^n \frac{|x_{ik} - y_{jk}|}{(x_{ik} + y_{jk})} \quad (12)$$

### 3.5.5 Distância de Canberra

Pode ser considerada mais uma modificação da distância *City Block*. Ocorre quando a soma das diferenças absolutas é normalizada pela diferença das coordenadas absolutas individualmente.

$$d_{ij} = \sum_{k=1}^n \frac{|x_{ik} - y_{jk}|}{|x_{ik}| + |y_{jk}|} \quad (13)$$

## 4 METODOLOGIA

### 4.1 BANCO DE DADOS

Na avaliação de um algoritmo para segmentação é necessário saber qual a segmentação perfeita. Em outras palavras, no contexto de nuvem de pontos, isso significa ter qual o agrupamento correto para cada ponto da PC, o que pode ser compreendido como gabarito da segmentação. A partir disso é possível avaliar métricas quantitativas como *v-measure*, completude, homogeneidade e acurácia.

A busca por bancos de dados que atendessem os critérios necessários para a pesquisa foi o maior obstáculo encontrado. Os critérios são:

- Ser um banco de dados sobre segmentação de nuvem de pontos;



- Preferencialmente segmentação não semântica;
- Ser um banco de dados *open source*;
- Possuir uma documentação básica;
- Possuir nuvem de pontos capturadas em indústrias ou para contexto industrial;
- Possuir nuvens com no máximo 200 mil pontos;
- Possuir uma certa variedade de nuvem de pontos.

Os primeiros dois pontos são triviais, contudo, o segundo poderia ser relevado após algum tratamento dos dados. Esse tratamento é mais intuitivo caso a documentação do banco de dados seja bem estruturada.

#### 4.1.1 Adaptação de banco de dados para segmentação não semântica

A grande maioria dos *datasets* (banco de dados) avaliados, tratava da segmentação semântica. As principais bases utilizadas para pesquisa de *datasets* foram o *paperswithcode* e o *kaggle*. O problema com a utilização do gabarito da segmentação semântica para a segmentação não semântica está na principal diferença entre elas, ou seja, as *labels* ou classes. Em nosso algoritmo, é esperado um valor numérico que delimita o seu grupo, nesse caso existiria o grupo zero, um, dois e assim por diante. Na segmentação semântica os grupos são *strings* (um valor textual) como, “carro”, “mesa”, “asa”, “motor” e etc. A conversão de um texto para um valor numérico é simples, o problema encontrado foi na documentação dos *datasets*. A organização de certos bancos *open source* não é muito elaborada. Muitas questões surgem devido a isso e para utilizar esses dados seria necessário dedicar um tempo para investigar e produzir testes. Outro aspecto encontrado na busca de banco de dados foram modelos de arquivos complexos que, provavelmente, estavam padronizados para o uso específico do responsável do *dataset*. Novamente, o tratamento desse banco de dados não era viável devido ao tempo disponível de desenvolvimento da pesquisa.

#### 4.1.2 Nuvem de pontos não industriais

Dentre os *datasets* explorados, a maioria não pertencia ao campo industrial. De outro modo, significa que não eram nuvem de pontos que descreviam peças, máquinas, tubulações, locais e estruturas. Contudo as *PCs* encontradas eram análogas a essas situações e poderiam ser utilizadas sem problemas. Além disso, dado que existisse uma

certa variação das *PCs* no que diz respeito a distância entre objetos, forma do objeto, espaço e definição dos objetos, o algoritmo poderia ser analisado sem maiores diferenças. Vale frisar que a definição de uma *PC* está ligada a quantidade de pontos por unidade de área que ela possui em sua superfície. Já que o dispositivo computacional disponível não é o suficiente para lidar com uma grande nuvem de pontos, existiu um limite na pesquisa diante desse aspecto. Como parâmetro de comparação, uma nuvem com 860 mil pontos é considerada grande para essa pesquisa. As nuvens utilizadas na pesquisa possuem cerca de 150 mil pontos.

#### 4.1.3 Decisões e possíveis pesquisas

Bancos de dados de alta qualidade foram encontrados, contudo, o tempo disponível para construção da pesquisa delimitou o uso deles em um momento futuro. Os principais datasets encontrados foram o *Shapenet*, a *Fusion360Gallery* e o *VASAD*. Deles, o *Fusion360Gallery* ofertado pela *Autodesk* não possui acesso open source e provavelmente deve ficar disponível após a aquisição do software da *Autodesk* que utiliza o dataset. O *VASAD: Volume and Semantic dataset for Building Reconstruction from Point Clouds*, é um banco de dados que possui vários prédios como escritório, vila, hospital e podem ser muito interessantes quando se trata da área de construção civil com automação. Entretanto, as nuvens de pontos são de maior dimensão do que o tratado na pesquisa. Já o *Shapenet* apresenta uma variedade enorme de objetos com gabaritos de segmentação semântica. Além da conversão necessária discutida anteriormente, nem todos os aspectos para lidar com esse banco de dados foram compreendidos. Um deles trata-se da formatação das *PCs*. Entretanto, este é um banco de dados *open source* com uma variedade importante para esse tipo de avaliação com algoritmos. Devido às limitações de desenvolvimento rápido da pesquisa atual esse banco de dados será explorado mais adiante em uma pesquisa futura. Por fim, o banco de dados utilizado trata-se do *EmergIMG*, que é um banco de dados de livre acesso disponibilizado pela USP (Universidade de São Paulo). Nele, existem nuvens de pontos de objetos, áreas e prédios suficientes para esse trabalho. Contudo, não possui o gabarito da segmentação não semântica. A implicação direta desse aspecto é que na análise de nuvem de pontos irá ocorrer uma avaliação visual qualitativa.

## 4.2 IMPLEMENTAÇÃO DO *K-MEANS*

Como explicitado anteriormente o *K-means* foi implementado pelo autor, enquanto o DBSCAN foi utilizado a partir de uma biblioteca *Python*. A implementação do *K-means*, foi feita a partir da linguagem de programação *Python* e auxílio da IDE (*Integrated Development Environment*), ou ambiente de desenvolvimento integrado, *PyCharm* da *JetBrains*. O código desenvolvido pode ser verificado na Figura 2.

Figura 2 - Implementação do *K-means* em *Python*.

```
1 import numpy as np
2 import open3d as o3d
3
4 # Carregamento de Arquivo da Nuvem de Pontos
5 input_file = "../PointsOfCloud/bumbameuboi.ply"
6 nuvem = o3d.io.read_point_cloud(input_file)
7
8 #Criação de Variáveis e escolha de valores
9 coordenadasDePlot = nuvem.points
10 coordenadas = np.asarray(nuvem.points)
11 tamanhoGrupal = 3
12 N = 2
13 training_set = coordenadas
14
15 # INICIALIZAÇÃO
16 a = np.random.randint(len(training_set), size=N)
17 codebook = np.zeros((N, tamanhoGrupal))
18 general_distortion = 0
19 iterations = []
20 partition = []
21 for i in range(0, N):
22     codebook[i] = np.copy(training_set[a[i]])
23
24 while True:
25     #PARTICIONAMENTO
26     min_index = []
27     for obj in range(len(training_set)):
28         dist = []
29         for i in range(len(codebook)):
30             dif = np.array(codebook[i]) - np.array(training_set[obj])
31             calc = np.linalg.norm(dif)
32             dist.append(calc)
33             min_index.append(dist.index(min(dist)))
34     partition = np.copy(min_index)
35
36     # TESTE DE PARADA
37     general_distortion = 0
38     for i in range(len(training_set)):
39         distortion = np.linalg.norm(training_set[i] - codebook[min_index[i]])
40         general_distortion += distortion
41     iterations.append(general_distortion)
42     if len(iterations)>2:
43         if (abs((iterations[-2]-iterations[-1])/iterations[-1]))< 0.001:
44             break
45
46     #ATUALIZAÇÃO
47     for i in range(len(codebook)):
48         centroid = []
49         for j in range(len(min_index)):
50             if i == min_index[j]:
51                 centroid.append(np.copy(training_set[j]))
52         if len(centroid) > 0:
53             codebook[i] = np.mean(centroid, 0)
54
55 #Aplicando cores a cada agrupamento
56 cores = np.zeros((N,tamanhoGrupal))
57 for i in range(N):
58     cor = np.random.random(3)
59     cores[i]=cor
60
61 for i in range(partition.shape[0]):
62     nuvem.colors[i] = cores[partition[i]]
63
64 o3d.visualization.draw_geometries([nuvem])
65
```

Fonte: O Autor (2022)

O código foi marcado de acordo com as etapas definidas na fundamentação teórica. Além disso, a Figura 2 mostra a versão específica do código para a distância euclidiana quadrática e a nuvem de pontos Bumbameuboi, Figura 3, do banco de dados EmergIMG.

Figura 3 - Nuvem de pontos Bumbameuboi.



Fonte: Nuvem de pontos retirada do banco de dados da USP EmergIMG.

Isso pode ser verificado na linha 30, 31 e 32 sobre a distância euclidiana e linha 5 sobre a PC. Entretanto, basta que sejam feitas breves alterações para que se modifique a distância ou a nuvem de pontos. Todas as distâncias abordadas anteriormente serão implementadas a partir da biblioteca *Python SciPy*. Ao fim do código o que temos é a visualização 3D da nuvem de pontos segmentada pelo algoritmo. Cada agrupamento é pintado com uma cor aleatoriamente escolhida como pode ser visto da linha 55 da Figura 2 e em diante. Por fim, o número de agrupamentos anteriormente referenciado por  $K$  é declarado no código como  $N$ . Portanto como  $N$  igual a 2, a nuvem de pontos será dividida em dois *clusters*. A resposta trazida pelo algoritmo para esses parâmetros acima é apresentada na Figura 4.

Figura 4 - Nuvem de pontos Bumbameuboi segmentada em dois *clusters*.



Fonte: Nuvem de pontos do banco de dados da USP EmergIMG.

Não é o objetivo da pesquisa aprofundar o uso das bibliotecas utilizadas na construção do algoritmo, logo, o funcionamento do código *Python* não será detalhado nesse trabalho.

### 4.3 UTILIZAÇÃO DO DBSCAN

Como frisado anteriormente, o algoritmo DBSCAN não foi elaborado pelo autor. As bibliotecas *Open3D* e *Scikit-Learn* foram utilizados para a análise desse algoritmo. A aplicação será exatamente como explicada anteriormente. O algoritmo irá segmentar a nuvem de pontos, pintar cada *cluster* com uma cor diferente e exibir a nuvem de pontos preenchida pelas cores.

### 4.4 TESTES E COMPARAÇÕES

#### 4.4.1 Teste 1: Comparação de algoritmos com nuvem de pontos 2D

Como explicado anteriormente, as análises com *Point Clouds 3D* não serão feitas de modo quantitativo devido ao fato de que não foi encontrado um *dataset* com um gabarito de segmentação. Contudo, é possível verificar quantitativamente os dois algoritmos utilizando dados fictícios o que permite entender as principais diferenças entre os algoritmos avaliados.

Na primeira etapa do teste, a partir da linguagem *Python*, foi gerado uma distribuição de pontos. O desvio padrão entre os pontos e o centro de seus respectivos *clusters* foi alterado a cada execução verificando a resposta dos algoritmos. Por fim, métricas como completude, homogeneidade e *V-measure* foram extraídos para os diferentes valores de desvio padrão. Em outras palavras, os modelos algorítmicos foram aplicados para a mesma distribuição de dados alterando-se as dispersões dos pontos a cada teste. Na segunda parte do teste, diferentes distribuições de dados foram segmentadas pelos dois algoritmos.

#### 4.4.2 Teste 2: Comparação visual de algoritmos com nuvem de pontos 3D

Neste teste aplicou-se os algoritmos à três nuvens de pontos. A bumbameuboi, Figura 3, a nuvem de pontos Guy, Figura 5 e a nuvem de pontos Quarto, Figura 6.

Figura 5 – Nuvem de Pontos Guy



Fonte: Nuvem de pontos retirada do banco de dados público *Sketchfab*

Figura 6 – Nuvem de Pontos Quarto



Fonte: Nuvem de pontos Retirada da biblioteca Python *Open3D*

O DBSCAN obtém como resposta o número de *clusters* estimado pelo algoritmo, já o *K-means* não. No algoritmo *K-means* é preciso definir como entrada a quantidade de grupos que o procedimento deve considerar. Então, alteramos o número de *clusters* gradualmente, no caso do *K-means*, até que ficasse equivalente visualmente com o DBSCAN.

#### 4.4.3 Teste 3: Comparação visual entre tipos de distância no *K-means*

Por fim, nesse último teste, foi modificado o tipo de distância utilizada no *K-means* mantendo fixo o número de *clusters*. Buscou-se o resultado visual que poderá ser visto adiante no capítulo de resultados. Apenas a nuvem de pontos Guy, Figura 5, foi utilizada nesse teste pois o número de agrupamentos dessa PC é mais claro, no caso são 2, o homem e a cadeira no fundo.

## 5 RESULTADOS E DISCUSSÃO

### 5.1 K-MEANS E DBSCAN

Dado que as características, estrutura e lógica dos algoritmos foram abordados, agora serão feitos teste comparativos entre eles utilizando diferentes distribuições de dados.

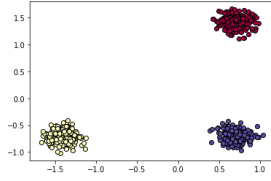
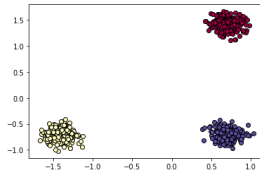
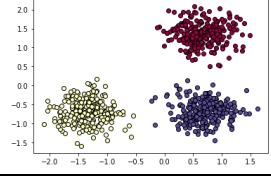
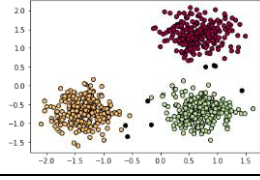
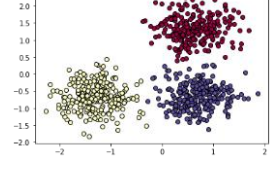
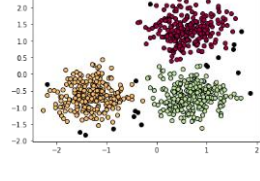
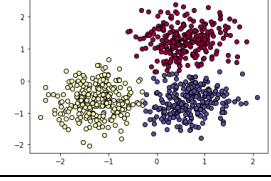
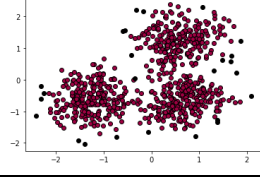
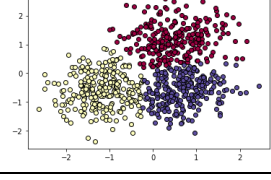
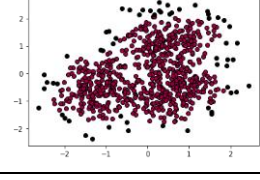
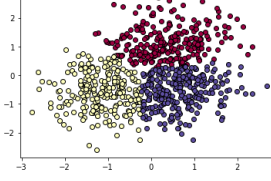
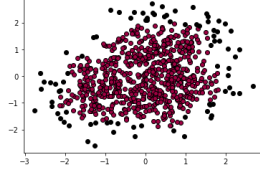
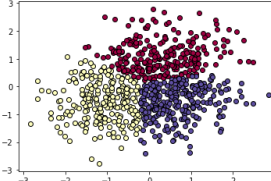
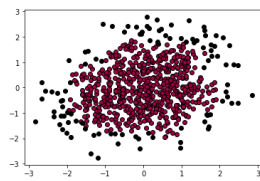
#### 5.1.1 Teste 1

Para simplificar a nomenclatura o desvio padrão das distâncias entre os pontos e os centros de seus respectivos *clusters* será denominado por dispersão ou valor de

dispersão. Quanto menor o valor de dispersão, mais agrupado são os pontos. A dispersão vai do infinito positivo à no mínimo 0 onde todos os pontos estão na mesma coordenada. A distribuição de pontos gerada possui 3 *clusters* e cada *cluster* possui uma localização chamada de centro. As coordenadas dos centros de cada *cluster* são (1,1), (1,-1) e (-1,-1). No total, são 750 pontos distribuídos da forma igualitária entre os 3 agrupamentos. Portanto, cada agrupamento possui 250 pontos. Além disso, cada grupo formado pelos algoritmos possui uma cor diferente, em particular ao DBSCAN, a cor preta é utilizada para pontos classificados como ruído. O valor de dispersão foi acrescido progressivamente de 0.1 até 1.1. A cada valor de dispersão adotado, a seleção visual feita pelo algoritmo e as métricas de avaliação foram registradas. Os resultados podem ser verificados no Quadro 2, Tabela 1 e Tabela 2.



Quadro 2 – K-means x DBSCAN

Dp	K-Means	DBSCAN
0.1		
0.3		
0.4		
0.5		
0.7		
0.9		
1.1		

Fonte: O Autor (2022)

Tabela 1 – Métricas avaliativas para o *K-means*

Dp	Homogeneidade	Compleitude	V-measure	Acurácia
0.1	1	1	1	1
0.3	1	1	1	1
0.4	0.945	0.945	0.945	0.989
0.5	0.872	0.872	0.872	0.971
0.7	0.716	0.716	0.716	0.920
0.9	0.519	0.520	0.520	0.839
1.1	0.393	0.394	0.393	0.775

Fonte: O Autor (2022)

Tabela 2 – Métricas avaliativas para o DBSCAN

Dp	Homogeneidade	Compleitude	V-measure	Acurácia	Nº de Ruídos	Grupos
0.1	1	1	1	1	0	3
0.3	0.989	0.949	0.969	0.989	8	3
0.4	0.953	0.883	0.917	0.972	18	3
0.5	0.001	0.007	0.002	0.319	25	1
0.7	0.007	0.029	0.012	0.295	58	1
0.9	0.008	0.024	0.012	0.281	84	1
1.1	0.012	0.032	0.017	0.268	104	1

Fonte: O Autor (2022)

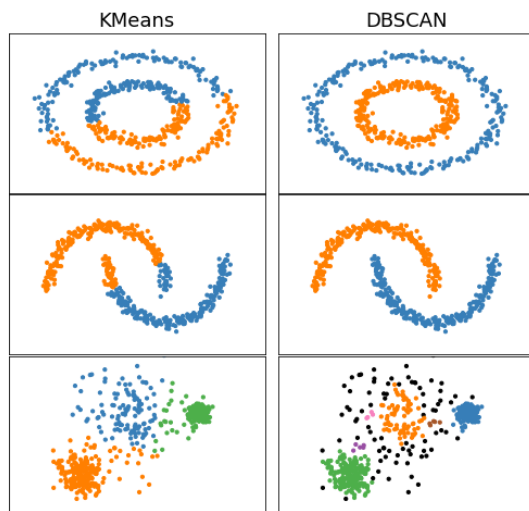
O Quadro 2 transmite visualmente a evolução dos algoritmos. Enquanto o *K-means* consegue lidar com o crescimento da dispersão no conjunto de dados, o DBSCAN passa por uma grande dificuldade em formar os agrupamentos corretos. Vale frisar que o *K-means* recebeu diretamente a informação de que há 3 agrupamentos, enquanto o DBSCAN possui a tarefa de definir quantos *clusters* existem.

A partir do desvio padrão maior do que 0.4 o DBSCAN passa a não conseguir diferenciar mais os agrupamentos, como se trata de um algoritmo baseado em densidade, o modelo acaba considerando toda a distribuição como um único *cluster*. Devido a isso, os valores de homogeneidade, completude, *V-measure* e acurácia caem

abruptamente. A acurácia passa a ser aproximadamente um terço quando a dispersão é de 0.5, o que é intuitivo já que existem 3 agrupamentos e o algoritmo só pôde acertar os pontos de 1 agrupamento, pois só definiu um *cluster*. Além disso, quanto maior a dispersão, maior é o número de pontos considerados ruídos pelo DBSCAN, chegando a somar 104 pontos ruidosos quando se trata da maior dispersão, como pode ser visualizado na Tabela 2. Ademais, é justamente a classificação dos pontos como ruído que faz com que o acerto do algoritmo caia abaixo de um terço. Já o *K-means*, possui um bom desempenho frente ao aumento da dispersão. Somente quando a dispersão é maior que 0.9 o algoritmo cai abaixo dos 84% de acerto. Este resultado possui um maior peso quando verificado que os pontos na dispersão de valor 1.1 já não podem ser, visualmente, distinguíveis e classificáveis. Ainda assim, o algoritmo possui um acerto de aproximadamente 78%.

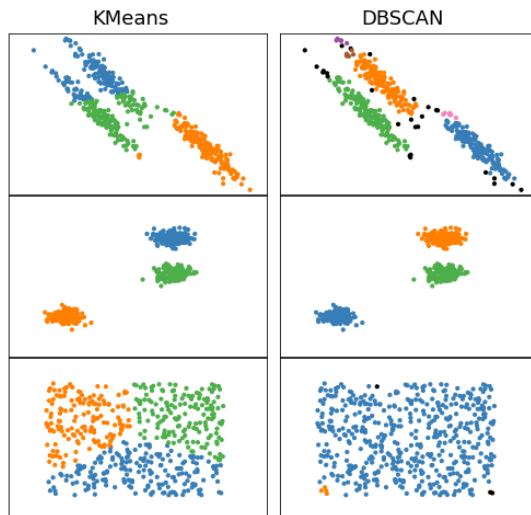
Em seguida diferentes tipos de distribuição de pontos foram elaborados, apresentadas na Figura 7 e na Figura 8. O ponto principal agora trata-se da configuração dos pontos e não da dispersão de cada *cluster*.

Figura 7 – Distribuição de dados A



Fonte: Nuvem de pontos 2D gerada a partir da biblioteca python *Scikit-Learn*

Figura 8 – Distribuição de dados B



Fonte: Nuvem de pontos 2D gerada a partir da biblioteca python *Scikit-Learn*

Diante dessas imagens fica mais claro as possíveis situações em que o *K-means* e o DBSCAN podem exercer seu melhor desempenho. Nas duas primeiras distribuições da Figura 7, o DBSCAN consegue definir sem problemas o número de *clusters* e segmentar com perfeição. Isto relaciona-se ao modelo de nuvem de pontos em que seus

agrupamentos estão separados por uma distância relevante. Logo, depende da forma que PC foi criada e das possíveis formas de digitalizar um objeto como nuvem de pontos. Como um bom exemplo na área industrial, tubulações poderiam ser segmentadas com perfeição pelo DBSCAN. Contudo, um fator do cenário real é que as tubulações nem sempre estarão igualmente distantes e não entrelaçadas. Para tal, uma possível solução seria a segmentação por partes, ou seja, sempre que as tubulações, partes e estruturas colidissem ou se entrecortassem, ficando muito próximos, a nuvem seria recortada e só então passaria pelo DBSCAN. Já o *K-means* também poderia segmentar tubulações, contanto que a definição da nuvem de pontos e a distância entre os *clusters* fosse suficiente para essa abordagem. O que geralmente não ocorre. Contudo, o *K-means* possui um melhor desempenho frente a nuvens de pontos com densidade constante e superfície única. A nuvem de pontos Bumbameuboi ilustra esse tipo de nuvem de pontos a qual o *K-means* possui melhor desempenho. Esta nuvem de pontos está descrita apenas na superfície, ou seja, não existem pontos dentro do corpo da nuvem. Dado que ela possui essas características, o DBSCAN delimitaria apenas 1 agrupamento, enquanto o *K-means* dividi a nuvem de acordo com número de *clusters* especificado.

### 5.1.2 Teste 2

Nesta seção foram observados a atuação dos algoritmos em 3 dimensões; para 3 nuvens de pontos. A ação do DBSCAN pode ser verificada na Figura 9, Figura 10 e Figura 11.

Figura 9 – Nuvem de Pontos Quarto Segmentada pelo DBSCAN



Fonte: O Autor (2022)

Figura 10 – Nuvem de pontos Bumbameuboi Segmentada pelo DBSCAN



Fonte: O Autor (2022)

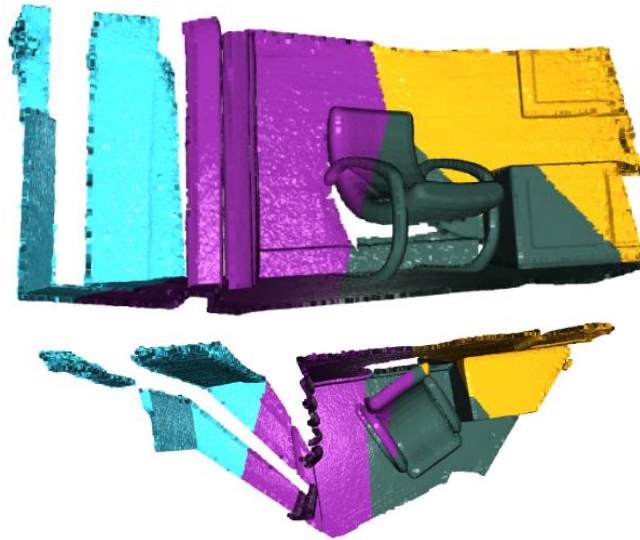
Figura 11- Nuvem de pontos Guy Segmentada pelo DBSCAN



Fonte: O Autor (2022)

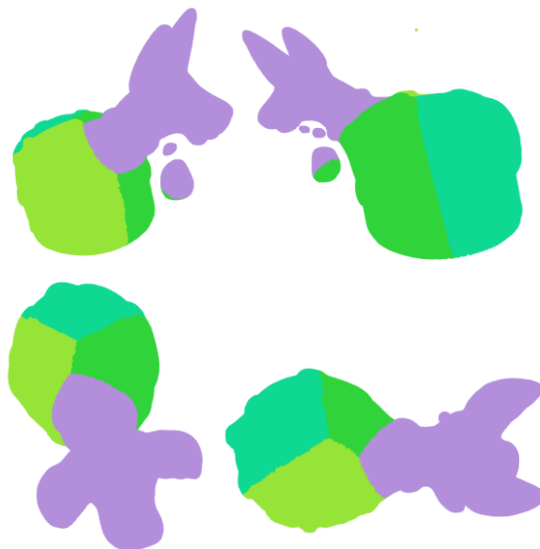
O *K-means* precisa da informação de quantos *clusters* devem ser formados. O procedimento foi fornecer ao *K-means* o número de *clusters* baseado na resposta visual do DBSCAN. Por exemplo na Figura 11, apesar do DBSCAN ter encontrado mais do que 2 *clusters*, existem 2 grupos principais. De igual modo, o número de *clusters* fornecido ao *K-means* foi 4 para a Figura 12, 4 para a Figura 13, e 2 para a Figura 14.

Figura 12 – Nuvem de pontos Quarto segmentada pelo K-means



Fonte: O Autor (2022)

Figura 13 – Nuvem de pontos Bumbameuboi segmentada pelo K-means



Fonte: O Autor (2022)

Figura 14 – Nuvem de pontos Guy segmentada pelo K-means







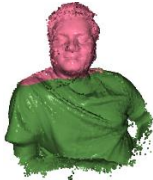





Fonte: O Autor (2022)

Nesse teste, o DBSCAN não só guiou o número de *clusters* que deveria ser utilizado pelo *K-means* como também conseguiu segmentar de forma mais precisa cada uma das nuvens exibidas. Enquanto isso, o *K-means* definiu seus *clusters* de forma aproximadamente simétrica e no geral não trouxe nenhuma segmentação com grandes benefícios. Exceto o caso da Figura 13 em que exatamente a cabeça do Bumbameuboi foi discernida do resto do corpo. Este resultado questiona a possibilidade do *K-means* ser utilizado para segmentar peças com partes muito bem definidas. Contudo, ao mesmo tempo existe uma imposição nessa possibilidade, que é a concentração de pontos em determinada parte da PC. A cabeça do Bumbameuboi foi definida como *cluster* devido ao seu tamanho e, principalmente, devido ao seu prolongamento para além do corpo. Outro fator diferencial é a sensibilidade que o *K-means* possui sobre a escolha dos pontos iniciais. Um método para aprimorar o *K-means* é entregar inicialmente onde cada *cluster* deve iniciar. Esse ato produz uma melhora significativa na resposta de segmentação do *K-means*, que não será abordado aqui neste trabalho, mas fica destacado como possível futura análise.

### 5.1.3 Teste 3

O *K-means* constrói seus agrupamentos a partir da regra do vizinho mais próximo, que é utilizada para encontrar a relação de proximidade entre os pontos com base em algum tipo de distância. Nesta seção, 5 diferentes tipos de distância foram avaliados para o *K-means*: a distância Euclidiana, Euclidiana quadrática, *City Block*, *Bray Curtis* e *Canberra*. Os resultados podem ser verificados no Quadro 3.

Quadro 3 -Avaliação de Distâncias

Distância	Segmentação com K-Means	
Euclidiana		
Euclidiana Quadrática		
City Block		
Bray Curtis		
Canberra		

Fonte: O Autor (2022)

A distância Euclidiana e Euclidiana quadrática não possuíram quase nenhuma diferença visual perceptível. A Euclidiana quadrática é justamente a utilizada no Teste 2 devido a esta ser a mais utilizada comumente. Já a distância de Bray Curtis demonstrou um resultado análogo as duas últimas citadas, contudo possui algumas oscilações nos seus limites entre os *clusters*. As distâncias de Canberra e *City Block*, definiram melhora região da cabeça onde estão muitos relevos e detalhes. Este resultado expõe a possibilidade de um estudo mais elaborado sobre essa distância em especial. O real impacto dessas distâncias não pode ser medido quando aplicado a apenas uma nuvem de pontos e ou a nuvem de pontos com aspectos parecidos. Contudo, é possível afirmar que essa nuvem de pontos não obteve melhora significativa com exceção das distâncias de Canberra e *City Block* que exprimiram certa relevância ao definir razoavelmente melhor a região da cabeça da nuvem de pontos *Guy*.



## 6 CONCLUSÃO

A segmentação é uma ferramenta essencial para muitos processos de reconhecimento visual feitos por máquinas autônomas, não só isso, mas também participa direta e indiretamente na área de automação industrial a partir da atuação com os irmãos digitais. O objetivo dessa pesquisa foi analisar técnicas e algoritmos de clusterização para segmentação no intuito de ampliar e organizar o conhecimento disposto por vários trabalhos. Mais especificamente, foram realizados testes para analisar quantitativamente e qualitativamente o desenvolvimento de cada algoritmo em frente a diferentes distribuições de dados e diferentes contextos. Além disso, o *K-means* foi o algoritmo mais detalhado e pôde ser averiguado no que concerne as diferentes distâncias disponíveis para o processo de busca do vizinho mais próximo. As distâncias estudadas não demonstraram impacto relevante na segmentação feita pelo *K-means*, para uma nuvem de pontos específica. Pequenas alterações ocorreram, mas nenhum fator conclusivo pôde ser declarado. Por fim, verificou-se que o *K-means* cumpre a melhor segmentação frente a um conjunto de dados com uma maior dispersão, enquanto o DBSCAN traz uma melhor definição de agrupamentos para nuvem de pontos que já possuem pelo menos uma breve distância bem definida entre seus principais segmentos. Em ambos os casos, caminhos para pesquisa e discussão foram sugeridos ao mesmo que o vínculo com a área de automação e controle sempre que possível foi introduzida para levantamento de hipóteses e sugestões.

## REFERÊNCIAS

- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 14724**: informação e documentação: trabalhos acadêmicos: apresentação. Rio de Janeiro: ABNT, 2011.
- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 6023**: informação e documentação - referências – elaboração. Rio De Janeiro: ABNT, 2018.
- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 6027**: informação e documentação — sumário — apresentação. Rio De Janeiro: ABNT, 2012.
- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 6028**: informação e documentação - resumo, resenha e revisão – apresentação. Rio De Janeiro: ABNT, 2021.
- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 10520**: informação e documentação - citações em documentos – apresentação. Rio De Janeiro: ABNT, 2002.
- AGAPAKI, Eva; BRILAKIS, Ioannis. CLOI-NET: Class segmentation of industrial facilities' point cloud datasets. **Advanced Engineering Informatics**, Cambridge, v. 45, n. 101121, 30 jun. 2020. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S1474034620300902> . Acesso em: 20 jun. 2022.
- BANDA, J.; ANGRYK, R.A. **Framework for creating large-scale content-based image retrieval system (cbir) for solar data analysis**. [S. l.], 2011. Disponível em: <https://www.cs.montana.edu/techreports/1011/Banda.pdf>. Acesso em: 10 set. 2022.
- BARGONI, A. *et al.* Competitive strategies in the agri-food industry in Italy during the COVID-19 pandemic: an application of K-means cluster analysis. **British Food Journal**, [s. l.], v. 124, n.12, 28 fev. 2022. DOI: <https://doi.org/10.1108/BFJ-07-2021-0738>. Disponível em: <https://www.emerald.com/insight/content/doi/10.1108/BFJ-07-2021-0738/full/html>. Acesso em: 10 mai. 2022.
- CHEN. *et al.* "KNN-BLOCK DBSCAN: Fast Clustering for Large-Scale Data". **IEEE Transactions on Systems, Man, and Cybernetics: Systems**, v. 51, p. 3939-3953, jun. 2021, doi: 10.1109/TSMC.2019.2956527. Disponível em: <https://ieeexplore.ieee.org/document/8936574>. Acesso em: 12 jan. 2022.
- CERUTTI, G. *et al.* A Model-Based Approach for Compound Leaves Understanding and Identification. **IEEE International Conference on Image Processing**, Melbourne, v.1, 14 out. 2013. Disponível em: [https://www.researchgate.net/publication/271482960\\_A\\_Model-Based\\_Approach\\_for\\_Compound\\_Leaves\\_Understanding\\_and\\_Identification](https://www.researchgate.net/publication/271482960_A_Model-Based_Approach_for_Compound_Leaves_Understanding_and_Identification). Acesso em: 20 jan. 2022.
- EMERGING image modalities representation and compression. [S. l.], [202-?]. Dataset. Disponível em: <http://uspaulopc.di.ubi.pt>. Acesso em: 10 jun. 2022.

ESTER, M. *et al.* A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. **2nd International Conference on Knowledge Discovery and Data Mining**. Portland, p. 226-231, 1996.

FAISAL, M.; ZAMZAMI, E. M.; SUTARMAN, S. Comparative Analysis of Inter-Centroid K-Means Performance using Euclidean Distance, Canberra Distance and Manhattan Distance. **Journal of Physics: Conference Series**, v. 1566, n. 012112, 2020. DOI: 10.1088/1742-6596/1566/1/012112. Disponível em: <https://iopscience.iop.org/article/10.1088/1742-6596/1566/1/012112>. Acesso em: 20 abr. 2022.

FONSECA, M. O. *et al.* Utilização de tecnologia 3d na automação. **17º Seminário de Automação**, São Paulo, p. 34-43, 2013. ISSN: 2594-5335, Disponível em: <https://abmproceedings.com.br/ptbr/article/utilizacao-de-tecnologia-3d-na-automao>. Acesso em: 21 abr. 2022.

FONSECA, C. S.; FERREIRA, F. A. B. S.; MADEIRO, F. Vector quantization codebook design based on fish school search algorithm. **Applied soft computing**, Recife, v. 73, p. 958-968, 29 set. 2018. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S1568494618305428>. Acesso em: 19 abr. 2022.

GEOGEBRA. **O que é o Geogebra?** [S. l.], [202-?]. Aplicativo. Disponível em: <https://www.geogebra.org/about>. Acesso em: 27 jun. 2022.

HE, Y. *et al.* Factors influencing carbon emissions from China's electricity industry: Analysis using the combination of LMDI and K-means clustering. **Environmental Impact Assessment Review**. Tianjin, v. 93, 22 dez. 2022. Disponível em: <https://doi.org/10.1016/j.eiar.2021.106724>. Acesso em: 30 abr. 2022.

HOW to use Kaggle. [S. l.], [202-?]. Dataset. Disponível em: <https://www.kaggle.com/docs/datasets>. Acesso em: 03 jun. 2022.

JUNG, H. Kyung.; KIM, H. K. Expansion of immersive experience in society. **Proceedings of the 1st international workshop on computational humanities and social sciences (Computing4Human 2020) CEUR Workshop Proceedings**, Seoul, ano 1, p. 22-32, 15 fev. 2020. Disponível em: <https://ceur-ws.org/Vol-2653/paper4.pdf>. Acesso em: 13 fev. 2022.

KARL. *et al.* Fusion 360 gallery: a dataset and environment for programmatic CAD construction from human design sequences. **ACM Trans. Graph**, New York, v. 40, n. 54, 19 jul. 2021. Disponível em: [https://dl.acm.org/doi/pdf/10.1145/3450626.3459818?casa\\_token=Q-VbZYRqPJMAAAA:9P39DDckae4CPR-MMHqhB4nkYRqk-PkTuSLxQkTghaKfxG7zgxKEiFwBqGBJQDbAJEJLDhamF9X\\_Yw](https://dl.acm.org/doi/pdf/10.1145/3450626.3459818?casa_token=Q-VbZYRqPJMAAAA:9P39DDckae4CPR-MMHqhB4nkYRqk-PkTuSLxQkTghaKfxG7zgxKEiFwBqGBJQDbAJEJLDhamF9X_Yw). Acesso em: 12 jun. 2022.

KUMAR, S.; SUHAIB, M.; ASJAD, M. Narrowing the barriers to Industry4.0 practices through PCA-FuzzyAHP-K means. **Journal Of Advances In Management Research**,

Nova Deli, v. 18, n.2, p. 200-226, 17 ago. 2020. Disponível em: <https://www.semanticscholar.org/paper/Narrowing-the-barriers-to-Industry-4.0-practices-Kumar-Suhaib/6d9cf688811422c315b423024b873e077c01d342>. Acesso em: 30 jul. 2022.

MEIRA, N. Edge AI – MASkRCNN e Segmentação de Instâncias. **IMobilis Computação Móvel**, Ouro Preto, 17 nov. 2020. Disponível em: <http://www2.decom.ufop.br/imobilis/segmentacao>. Acesso em: 6 jun. 2022.

MORAES, M. B. C. **Análise Multivariada Aplicada à Contabilidade**. São Paulo, 2016. Disponível em: [https://edisciplinas.usp.br/pluginfile.php/2232110/mod\\_resource/content/1/AnáliseMultivariada-Aula12.pdf](https://edisciplinas.usp.br/pluginfile.php/2232110/mod_resource/content/1/AnáliseMultivariada-Aula12.pdf). Acesso em: 05 set. 2022.

PEDREGOSA, F. *et al.* Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, [s. l.], n. 12, p. 2825-2830, 2 jan. 2012. Disponível em: <https://arxiv.org/pdf/1201.0490.pdf>. Acesso em: 13 abr. 2022.

OUR Mission. [S. l.], [202-?]. Dataset. Disponível em: <https://paperswithcode.com/about>. Acesso em: 03 jun. 2022.

RIBEIRO, J. *et al.* Robotic Process Automation and Artificial Intelligence in Industry 4.0 - A Literature review. **Procedia Computer Science**, Viana do Castelo, Portugal, v. 181, p. 51-58, 22 fev. 2021. Disponível em: <https://reader.elsevier.com/reader/sd/pii/S1877050921001393?token=8FE2AC635B5B8380D3DF90C31B6CE45BB2ACB2158E9B9587EB5B4BD8EB8061D6025D765AA3F9A5F2FE4F54BAB2E75721&originRegion=us-east-1&originCreation=20230205180419>. Acesso em: 24 ago. 2022.

ROSENBERG, A.; HIRSCHBERG, J. V-Measure: A conditional entropy-based external cluster evaluation measure. **Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)**, Prague, v.1, p. 410-420, 2007. Disponível em: <https://aclanthology.org/D07-1043.pdf>. Acesso em: 15 mai. 2022.

SCIKIT-LEARN. **Clustering**. [S. l.], [202-?]. Website. Disponível em: <https://scikit-learn.org/stable/modules/clustering.html>

SHAPENET. **What is ShapeNet?** [S. l.], [202-?]. Dataset. Disponível em: <https://shapenet.org/about>. Acesso em: 23 jun. 2022.

SAMPATH, A.; SHAN, J. Clustering based planar roof extraction from lidar data. **ASPRS 2006 Annual Conference Reno**, Nevada, 1 mai. 2006. Disponível em: [https://engineering.purdue.edu/~jshan/publications/2006/ASPRS\\_2006\\_Lidar.pdf](https://engineering.purdue.edu/~jshan/publications/2006/ASPRS_2006_Lidar.pdf). Acesso em: 12 jun 2022.

SU, Y.; BETHEL, J.; HU, S. Octree-based segmentation for terrestrial LiDAR point cloud data in industrial applications. **ISPRS Journal of Photogrammetry and Remote Sensing**, West Lafayette, v. 113, p. 59–74, 2016. Disponível em:

<https://www.sciencedirect.com/science/article/abs/pii/S0924271616000022>. Acesso em: 15 jun. 2022.

SUWILO, S.; PULUNGAN, A. F.; ZARLIS, M. Analysis of Braycurtis, Canberra and Euclidean Distance in KNN Algorithm. **Journal Publications & Informatics Engineering Research**. Medan, v.4, 1 out. 2019. Medan, Indonesia. Disponível em: <https://jurnal.polgan.ac.id/index.php/sinkron/article/view/10207/257>. Acesso em: 19 jun. 2020.

VASAD: A Volume And Semantic Dataset For Building Reconstruction From Point Clouds. [S. l.], [202-?]. Dataset. Disponível em: <https://github.com/palanglois/vasad>. Acesso em: 10 jun. 2022.

WANG, Q.; KIM, M. Applications of 3D point cloud data in the construction industry: A fifteen-year review from 2004 to 2018. **Advanced Engineering Informatics**, Singapore, v. 39, p. 306-319, 15 fev. 2019. ISSN 1474-0346. Disponível em: <https://doi.org/10.1016/j.aei.2019.02.007>. Acesso em: 10 jun. 2022.

WHAT is LIDAR? [S. l.], 26 fev. 2021. Disponível em: <https://oceanservice.noaa.gov/facts/lidar.html>. Acesso em: 12 jul. 2022.

XIE, Y.; TIAN, J.; ZHU, X. X. Linking Points With Labels in 3D: A Review of Point Cloud Semantic Segmentation. **IEEE Geoscience and Remote Sensing Magazine**, Munich, v. 8, n. 4, p. 38-59, 2020. Disponível em : <https://ieeexplore.ieee.org/document/9028090>. Acesso em: 25 fev. 2022.

XU, Y. *et al.* Introduction to point cloud compression. **ZTE Communications**, Shanghai, v. 16, n. 3, p. 8, 24 ago. 2018. Disponível em: [https://res-www.zte.com.cn/mediares/magazine/publication/com\\_en/article/201803/XUYiling.pdf](https://res-www.zte.com.cn/mediares/magazine/publication/com_en/article/201803/XUYiling.pdf) . Acesso em: 21 mar. 2022.

YUSUF, B. **Everything you need to know about scan-to-BIM**. [S. l.], 5 mai. 2020. Disponível em: <https://www.navvis.com/blog/everything-you-need-to-know-about-scan-to-bim> . Acesso em : 07 ago. 2022.

YIN, C. *et al.* Automated semantic segmentation of industrial point clouds using ResPointNet++. **Automation in Construction**, v.130, 10 ago. 2021. DOI: 10.1016/j.autcon.2021.103874. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S0926580521003253>. Acesso em: 10 ago. 2022..

ZERMAS, D.; IZZAT, I.; PAPANIKOLOPOULOS, N. Fast segmentation of 3D point clouds: A paradigm on LiDAR data for autonomous vehicle applications. (**IEEE International Conference on Robotics and Automation (ICRA)**), Singapore, p. 5067-5073, 2017. DOI: 10.1109/ICRA.2017.7989591. Disponível em: <https://ieeexplore.ieee.org/document/7989591>. Acesso em: 10 ago. 2022.

ZHOU, Q.; PARK, J.; KOLTUN, V. **Open3D: A Modern Library for 3D Data Processing**. [S. l.], 30 jan. 2018. Disponível em: <https://arxiv.org/pdf/1801.09847.pdf> . Acesso em: 30 mai 2022.

ABOUT zivid. [S. l.], [202-?]. Site. Disponível em: <https://www.zivid.com/about> .  
Acesso em: 02 mar. 2022.