algorithms

MDPI

Article

CUDA and OpenMp Implementation of Boolean Matrix Product with Applications in Visual SLAM

Amir Zarringhalam [1], Saeed Shiry Ghidary [1,2,*], Ali Mohades [1] and Seyed-Ali Sadegh-Zadeh [2,*]

[1] Computer Science and Mathematics Department, Amirkabir University of Technology, 424 Hafez Ave, Tehran 15916634311, Iran
[2] Department of Computing, Staffordshire University, Stoke-on-Trent ST4 2DE, UK
* Correspondence: shiry@aut.ac.ir (S.S.G.); ali.sadegh-zadeh@staffs.ac.uk (S.-A.S.-Z.)

Abstract: In this paper, the concept of ultrametric structure is intertwined with the SLAM procedure. A set of pre-existing transformations has been used to create a new simultaneous localization and mapping (SLAM) algorithm. We have developed two new parallel algorithms that implement the time-consuming Boolean transformations of the space dissimilarity matrix. The resulting matrix is an important input to the vector quantization (VQ) step in SLAM processes. These algorithms, written in Compute Unified Device Architecture (CUDA) and Open Multi-Processing (OpenMP) pseudo-codes, make the Boolean transformation computationally feasible on a real-world-size dataset. We expect our newly introduced SLAM algorithm, ultrametric Fast Appearance Based Mapping (FABMAP), to outperform regular FABMAP2 since ultrametric spaces are more clusterable than regular Euclidean spaces. Another scope of the presented research is the development of a novel measure of ultrametricity, along with creation of Ultrametric-PAM clustering algorithm. Since current measures have computational time complexity order, $O(n^3)$ a new measure with lower time complexity, $O(n^2)$, has a potential significance.

Keywords: ultrametricity; dissimilarity spaces; Fast Appearance Based Mapping; Open Multi-Processing

Citation: Zarringhalam, A.; Shiry Ghidary, S.; Mohades, A.; Sadegh-Zadeh, S.-A. CUDA and OpenMp Implementation of Boolean Matrix Product with Applications in Visual SLAM. *Algorithms* **2023**, *16*, 74. https://doi.org/10.3390/a16020074

Academic Editor: Vangelis Th. Paschos

Received: 28 October 2022
Revised: 14 December 2022
Accepted: 10 January 2023
Published: 29 January 2023

1. Introduction

Advances in robotics and artificial intelligence have enabled the creation of autonomous systems and vehicles (AV). AVs are expected to localize themselves, create maps, and navigate through unknown environments. These major tasks are typically achieved through simultaneous localization and mapping (SLAM). It is attempted in this study to use and combine the concept of ultrametricity in the SLAM process. Clustering is critical in the Vector Quantization (VQ) step of the SLAM process for loop closure detection because it helps to reduce the cumulative error of the robot's estimated pose and generate a consistent global map. SLAM algorithms have several sub-problems, the most important of which is loop closure detection (LCD). In this paper, we look at how clustering using ultrametric distance matrices affects the LCD part of the SLAM. Subdominant ultrametric distances are calculated by performing the transformations described in [1–3] on Euclidean distance matrices. The primary reason for using ultrametric spaces is that they can improve clustering algorithms by generating more compact clusters with a higher Donn index, which is calculated as a ratio of the smallest within-cluster distance to the largest between-cluster distance; a higher Donn index indicates better clustering. Ultrametric spaces are high-dimensional Hilbert spaces with a completely hierarchical structure. These spaces offer several computational benefits. For example, using an ultrametric configuration of the space can improve searching. The ability to structure data as a tree, as illustrated by our observations embedded in an ultrametric topology, provides a significant advantage for proximity searching. If we preprocess the data using such an embedding, we can find an observation's nearest neighbor in constant computational time, i.e., O(1) time, [4–7].

In order to create clusters, we apply the PAM algorithm [8] which searches a data set for k representative objects (K Medoids) and assigns each object to the closest Medoid. The overall goal is to minimize the sum of dissimilarities between cluster objects and the cluster's center (Medoid) [9]. One of the advantages of PAM clustering algorithm is that it is less sensitive to outliers than the other partitioning algorithms [10]. In this work, we present a new Ultrametric-PAM and incorporate it into the LCD pat of the Fast Appearance Based Mapping (FABMAP) SLAM procedure. The regular FABMAP [11], is available online at [12]. It significantly improves accuracy and speeds up computations by a factor of 1000.

Furthermore, we introduce a new measure of ultrametricity and demonstrate that the new measure significantly reduces the time complexity from order $O(n^3)$ to $O(n^2)$ in several different algorithms.

We parallelize the serial algorithm presented in Section 5, which can be embedded in the VQ part of FABMAP algorithm, to scale up our proposed approach to large data sets. For real-world-size datasets, parallelization of set of transformations is required because applying these transformations serially is nearly intractable. We present two different implementations of this algorithm, with Compute Unified Device Architecture (CUDA) and Open Multiprocessing (OpenMP).

The following is how the paper is structured. Section 2 is devoted to a review of previous works in this field as well as the motivation for introducing a new measure of ultrametricity. The set of transformations on the Euclidean distance matrices defined in [1] are then briefly reviewed in Section 3. Sections 4 and 5 define the Ultrametric-FABMAP SLAM method, introduce the new measure, and describe the parallel algorithms used to compute the transformations. In Section 6 we present the necessary definitions as well as the notion of ultrametric space, as well as motivating examples. To evaluate the results of the Ultrametric-PAM algorithm, we run simulation experiments on large-scale datasets. Furthermore, we compare the presented measure with Rammal's measure [13].

2. Prior Works

Several solutions to the LCD problem have been proposed in recent years, the majority of which are based on odometry, which uses ultrasonic sensors or lasers. Advances in computer vision and image processing, as well as widespread use of digital cameras, have accelerated the adoption of SLAM methods. Several deep learning-based solutions have been developed. LCD-Net [14], identifies concurrently visited locations and computes the 6-Dof transformation matrix to map loop closures in LiDAR point clouds. A global descriptor extraction block, a shared encoder and a relative pose head construct the transformation matrix between the point clouds in this model. Such customized deep learning methods, however, fail when the robot revisits previous locations with a significantly different perspective. DeepRING [15] solves this problem by learning a rotation-translation invariant representation from LiDAR scan, which prevents performance degradation due to extreme changes in the local view.

In [16,17], an autoencoder (AE) and a hyperbolic graph convolutional neural network (HGCN) were used in the vector quantization step of the FAB-MAP algorithm. For the clustering task, the AE-FABMAP model employs a self-supervised module, which can be thought of as a type of unsupervised algorithm. This approach requires significantly less memory (1/100) than the alternative semi-supervised approach implemented in HGCN-FABMAP. This significant reduction in memory requirements motivated us to investigate unsupervised clustering methods further.

During the 1990s, hierarchical clustering algorithms were among the best options. These methods, however, are not scalable because their parallelization requires keeping the entire dataset in memory, effectively limiting their applicability to datasets with fewer than 500,000 items [18]. Murtagh [18] developed a technique to avoid this problem by using ultrametrization via Baire space embedding [19–21]. For examples of Baire space please look at [18].

There has been a great deal of research into measuring the ultrametricity of dissimilarity spaces. A dissimilarity space is a set of points along with a function defined on them with properties of positiveness, symmetry, and reflectivity. An ultrametric space is a dissimilarity space with a strong triangular inequality as an additional condition. As an example of an ultrametricity measure, Rammal [13] defines the ultrametricity index as follows:

$$R = \sum_{x,y \in spaceX} \left(d(x,y) - d_c(x,y)\right) \Big/ \sum_{x,y \in spaceX} d(x,y) \tag{1}$$

where d and d_c are the Euclidean and the Cophenetic distance matrices of the space X. The $d_c(X,Y)$ is the proximity at which X and Y are grouped into the same cluster. Please note that $R \in [0,1]$. For instance, if X is an ultrametric space, $R(X) = 0$ and if X is a non-hierarchical space, $R(X) = 1$. For a more in-depth discussion of Rammal's measure, see [13].

Murtagh introduced a new ultrametricity index based on the shape and proportion of triangles in the space in [22] where triangles that are "almost" isosceles form a more hierarchical data space. For real-size datasets, computing Murtagh's measure is intractable because counting the number of triangles in a given space with n points is of order $O(n^3)$. As a result, we need to define a new measure based on the shape of the triangles in the space. The following section provides a brief overview of the transformations that increase the hierarchiality of a data space. The transformations are required as input to PAM clustering, which is a critical module in the VQ step of FABMAP.

3. Review of the Transformation

In what follows, we introduce a set of transformations defined by [1], on the Euclidean distance matrix A, that improves the hierarchical structure of dissimilarity space. These transformations are in the form of powers of Euclidean distance matrices over a lattice structure.

Let $\mathbb{M}(\mathbb{L})$ be a set of square matrices over a distributive lattice L, then for any $A_1, A_2, A_3 \in \mathbb{M}(\mathbb{L})$.

$A_1 * A_2$ is defined as:

$$\bigvee_{l=1}^{n} (a_{il} \wedge b_{lj}) \quad for\ 1 \leq i,j \leq n \tag{2}$$

The $\bigvee$ and $\bigwedge$ operators are defined as follows:

$$\bigvee_{l=1}^{n} (X_1, X_2, \ldots, X_n) = min\{X_1, X_2, \ldots, X_n\} \tag{3}$$

$$\bigwedge_{l=1}^{n} (X_1, X_2, \ldots, X_n) = max\{X_1, X_2, \ldots, X_n\} \tag{4}$$

Let the identity element be:

$$I_n = (\delta_{ij}) \begin{cases} 1 & \text{i = j} \\ 0 & \text{i} \neq \text{j} \end{cases} \tag{5}$$

Then from the definition of $\bigvee$ and $\bigwedge$:

$$\begin{cases} A_1 * (A_2 * A_3) = (A_1 * A_2) * A_3 \\ A_1 * I = I * A_n \\ A^{i+1} = A^i * A \end{cases} \tag{6}$$

Remark 1. *For all $A \in \mathbb{M}(\mathbb{L})$, the sequence $A^1, A^2, \ldots, A^n, \ldots$ is ultimately periodic, because it contains a finite number of distinct matrices. This sequence forms a cyclic group in terms of multiplication [1].*

The relation $\leq$ between any two matrices A^1, A^2 from set $\{A^1, A^2, ..., A^n\}$ is defined as follows:

$$A^1, A^2 \iff [A]_{ij} \geq [A^2]_{ij} \quad \forall i, j \in [1 \ldots n] \tag{7}$$

Theorem 1. *Let $A \in M_n$, with the property $A \leq A^2$, then A converges to $A^{k(A)}$ with $k(A) \leq n$.*

For proof, see Theorem 5.1 in [1].

4. The New Ultrametricity Measure

We propose a new measure of ultrametricity based on the deviation of large triangles from the closest isosceles triangles. The following definitions are for large triangles that form the general structure of the space. To define the term large triangle, we first ascendingly sort the distances between data points in the Euclidean distance matrix. Then, for each of the three consecutive triplets of distances that satisfy the triangular inequality, we choose one. The triangle they form is compared to the closest isosceles triangle by subtracting the largest side from the second largest side. The smaller the difference, the closer the triangle is to the isosceles one.

$$\sum_{\substack{i \in [0,(n \times n-1)/2] \\ \text{Tringular}(d_i, d_{i+1}, d_{i+2})}} \frac{1}{|d_i - d_{i+1} - \epsilon|} \tag{8}$$

In (8), each d_i refers to a triangle side. For the isosceles and equilateral cases, the parameter $0 < \epsilon < 1$ is included in the denominator. For example, consider the distances 3, 2, 1.5 that form triangle A and 2, 2, 1.5 that form triangle B, and $\epsilon = 0.001$. For triangle A the new measure will be $1/|2 - 2 + 0.001|$ which is larger than $1/|3 - 2 + 0.001|$ and at the end these values should be normalized.

5. Ultrametric-PAM and Ultrametric-FABMAP

In this section, we describe a set of serial and parallel algorithms(Algorithms 1–4) to transform an Euclidean distance matrix into a hierarchical distance matrix.

The standard Ultrametric-PAM algorithm will be discussed first. It is desired to investigate how compactness, Donn index, and other clustering evaluation criteria change when higher ultrametric configuration dissimilarity matrices are used as input to the PAM clustering algorithm.

Algorithm 1 determines how well PAM clustering algorithm identifies clusters in a dataset by evaluating clustering compactness, Donn index, sum of squared errors, and normalized mutual information (NMI), for dissimilarity spaces $(X, A^1), (X, A^2), \ldots, (X, A^k(A))$. The NMI, compactness, sum of squared error, and Donn index criteria are measured. One of the algorithm's potential benefits is that it can detect spiral clusters, which most clustering algorithms that are not density-based fail to detect. However, Algorithm 1 can only be used on toy datasets; for real-world-size datasets, this algorithm takes a long time. As a result, it must be parallelized. A possible parallel version can be achieved using OpenMp and Intel intrinsic Advanced Vector Extension (AVX-256) functions. Algorithm 2 is designed to speed up the transformations even further.

Another option is to parallelize the slow serial Algorithm 1 using GPU programming. Algorithm 3 presents the proposed solution.

Algorithm 3, is the CUDA-C implementation of Algorithm 1. In Section 6 we compare Algorithm 3 to Algorithm 2 and show that by distributing the program on computer processors and using Intel intrinsic modules, we can achieve superior performance that is even faster than GPU implementation. As shown in Algorithm 4, the resulting clustering is then incorporated into the FABMAP SLAM process.

Algorithm 4 generates the cluster centroids and BoW representation of the training and testing data required by the FABMAP algorithm. To train the Chow-Liu tree in the FABMAP algorithm, we use the BoW representation and the clustering centroid of the training data set. In the next section, we present the clustering evaluation metrics used in performance benchmarks.

Algorithm 1 (Ultrametric-PAM) Runs PAM with the transformed distance matrices

Input: Euclidean distance matrix A
Output: matrix of a dendrogram

procedure ULTRAMETRIC-PAM($K \leftarrow 1$, $B \leftarrow A$, $S \leftarrow \{\}$, $C = n \times n$)
  while $A^K \neq A^{K+1}$ **do**
    $PAM(A^K, n) / PAM(A^K, n)$
    err(K) = Sum of squared errors
    Calculate Clusterings Compactness and Donn index
    for $i = 1$ to n **do**
      for $j = 1$ to n **do**
        for $t = 1$ to n **do**
          Add $Max(A[i,t], B[t,j]))$ to S
        end for
        $C[i,j] \leftarrow Min(S)$
        Make S empty
      end for
    end for
    $A \leftarrow C$
    $K \leftarrow K + 1$
  end while
  Return A
end procedure

Because the algorithm is a very slow serial procedure, it must be parallelized before it can be used on real-world-sized datasets. For this purpose, we propose two different algorithms.

Algorithm 2 AVX-256 Transformation

```
omp_set_num_threads(80);
# pragma omp parallel for
for (int = 0; l < rows; l++)
    for (int p = 0; p < rows; p++)
        float *a, *b,copyC;
        _m256 *aAVX, *bAVX, *cAVX;
        posix_memalign((void **)& copyC, 32, Size);
        posix_memalign((void **) & a, 32, Size);
        posix_memalign((void **) & b, 32, Size);
        a = Distance_matrix[l]; b = Distance_matrix^T[l]
        aAVX = (void*)a; bAVX = (void*)b; cAVX = (void*) copyC;
        for (int g=0; g < length; g += 8, aAVX++, bAVX++, cAVX++)
            *cAVX = _mm256 _max _ps(*aAVX, *bAVX);
        result[l][p] = AVX_find_peaks(copyC,lenth);
```

Algorithm 3 CUDA-TRANSFORMATION

```
__global __void CUDA-Transformation(float *M, float *N, float *P){
    int tx = blockIdx.x * blockDim.x + threadIdx.x;
    int ty = blockIdx.y * blockDim.y + threadIdx.y;
    Minimum = large number;
    If (tx < blockDim.x & ty < blockDim.y )
        for k = 0 to WIDTH
            If (fmax (M[ty* WIDTH +k],N[k* blockDim.x +tx]) ≤ Minimum)
                Minimum =fmax(M[ty* WIDTH +k],N[k* blockDim.x +tx])
    P[ty* blockDim.x +tx] = Minimum
```

Algorithm 4 (Ultrametric-FABMAP). Runs FABMAP with the transformed distance matrices

Input: Euclidean distance matrix A, Training, Testing Dataset
Output: confusion matrix

procedure ULTRAMETRIC-FABMAP(Training, Testing Images)
    Extract 128-d surf features from all collected images
    Both for tranining and testing dataset
    Apply PCA on the training and testing images matrix

    Centroids ←Call Ultrametric-PAM for the feature matrix

    for $\mathcal{X} \in$ {Train/Test} **do**
        for All images $\in \mathcal{X}$ **do**
            Vector quantize image descriptors
            set on the obtained centroid
            Perform TF-IDF Vector Quantization
            Obtain BoW representation of of image set $\mathcal{X}$
        end for
        Normalize BoW representation, Obtained
        Start FABMAP procedure using Clustering centroids and BoW representaion
    end for
    Return confusion matrix
end procedure

6. Experiments

6.1. Metrics

For clusterings $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \ldots, \mathcal{C}_n\}$ where $\mathcal{C}_i$ is the ith cluster:

The Dunn Index is calculated as the ratio of the smallest distance between observations that are not in the same cluster to the greatest intra-cluster distance. Donn index is defined as follows:

$$DonnIndex(\mathcal{C}) = \frac{\min\limits_{C_k, C_l \in \mathcal{C}, C_k \neq C_l} (dist(i,j))}{\max\limits_{C_m \in \mathcal{C}} diam(C_m)} \tag{9}$$

A higher Dunn Index will indicate compact, well-separated clusters, while a lower index will indicate less compact or less well-separated clusters. We can now try to calculate the metric for the dataset we've created previously. Another clustering evaluation technique that we have used is compactness which can be defined as:

$$Compactness = \sum_{\mathcal{C}_i \in \mathcal{C}} \frac{1}{|\mathcal{C}_i| - 1} \sum_{x \in \mathcal{C}_i - \{m_i\}} d(x, m_i) \Big/ |\mathcal{C}| \tag{10}$$

Compactness determines how close the data points within a cluster are, to the cluster's Medoid. Formula adds the distances between data points of a clusters and each cluster's

Medoid, m_i. A lower within-cluster variation indicates a good compactness (i.e., good clustering).

6.2. Definition and Examples:

A dissimilarity space ($\mathcal{X}$,$\mathcal{F}$), is a pair where $\mathcal{X}$, is a set of points and $\mathcal{F}$:$\mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{R}$ is a single-valued distance function defined on them with the following three properties:

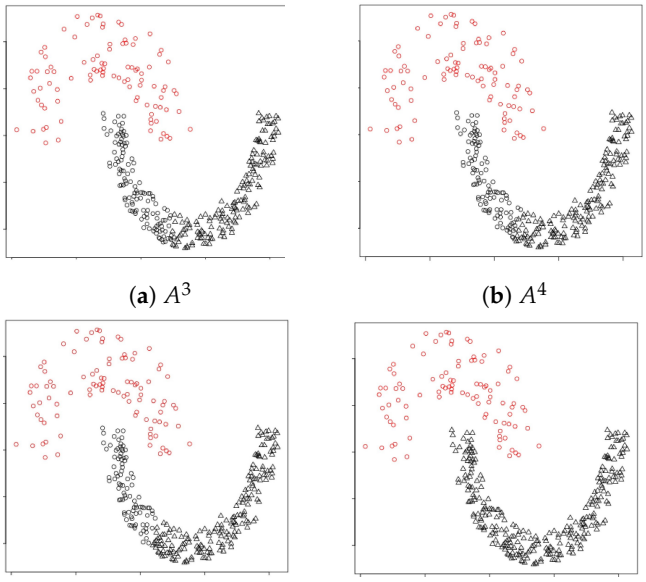
$$\begin{cases} \mathcal{F}(x,x) = 0 \\ \mathcal{F}(x_1,x_2) \geqslant 0 \text{ (positiveness)} \\ \mathcal{F}(x_1,x_2) = \mathcal{F}(x_2,x_1) \text{ (symmetry) for all } x_1, x_2 \in \mathcal{X}. \end{cases} \tag{11}$$

An ultrametric space is a dissimilarity space with additional conditions $F(x_1, x_2) = 0$, if and only if $x_1 = x_2$, and $F(x_1, x_3) \leq maxF(x_2, x_3), F(x_1, x_2)$. This inequality is called the strong triangular or ultrametric inequality. In this case, every triple of points in an ultrametric space forms an isosceles triangle.

6.3. Datasets

We applied Algorithm 1 on data sets Jain, Spiral, Iris, and Seeds. These datasets are available online (https://archive.ics.uci.edu/ml/index.php, accessed on 20 December 2017). Jain is a dataset including 373 two-dimensional points divided into two clusters. The Spiral dataset contains 300 data points and is made up of three spiral shape clusters in two-dimensional space. The Iris dataset contains three classes of 50 instances each, and each class represents a different type of plant. The Seeds dataset contains 210 data points and consists of kernels from three different wheat varieties. We discovered that increasing ultrametricity increased the Donn index of the clusters. Also, the Spiral dataset's compactness improves, but for the datasets Jain, Iris, and Seeds, lowering the ultrametric configuration improves compactness. We noticed that sum of squared error decreases with increasing ultrametricity for data sets Iris and Seed, while normalised mutual information improves for datasets Jain and Spiral.

Figure 1 shows that the Ultrametric-PAM algorithm correctly identifies clusters in the Jain dataset while increasing the ultrametricity of the dissimilarity space (X, A).



(**a**) A^3 (**b**) A^4

(**c**) A^5 (**d**) A^{32}

Figure 1. Identifying clusters in Jain dataset using PAM clustering algorithm as we apply the transformation discussed in Section 5 repeatedly.

Figure 2 shows the three clustering evaluation metrics Compactness, Donn index, and NMI increasing for the Jain dataset as the ultrametricity of the dataset's dissimilarity matrix increases. Figure 2 depicts how these metrics change in general, and Table 1 shows the numerical values of these criteria at each iteration of Algorithm 1.

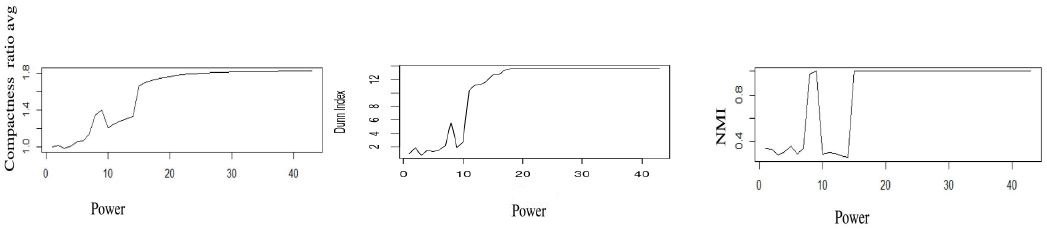


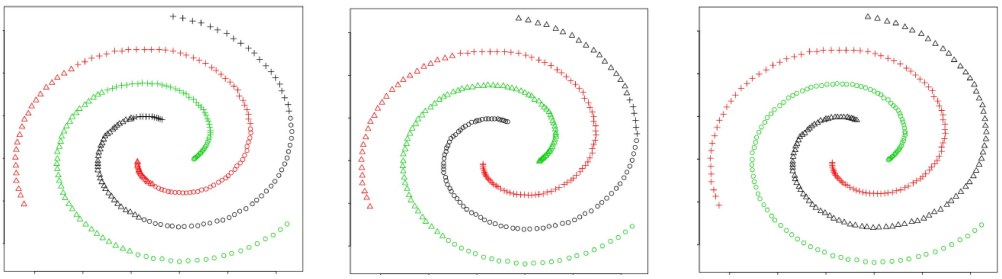
Figure 2. Compactness, Donn index and Normalized mutual information. Jain dataset.

According to Table 1, increasing the ultrametricity of the dissimilarity matrix of the Jain dataset results in clusterings with a better Donn index and higher compactness variation.

Table 1. Measuring compactness and Donn index as increase ultrametricity. Jain data set.

	Compactness Avg.	**Compactness Ratio Avg.**	**Dunn Index**	**Dunn Index Ratio**
B^3	0.4826358111	1.0163461701	0.0246030	0.7697904
B^4	0.4663649138	0.9820825208	0.04667035	1.460238898
B^5	0.4799552411	1.0107013609	0.0418983	1.310931155
B^6	0.5017430685	1.056582695	0.047757	1.494265365
B^8	0.5400191638	1.1371854226	0.1781176	5.5730085
B^{32}	0.8553148705	1.8164010431	043437224	13.5907955

Figure 3 shows that as the ultrametricity of the dataset's dissimilarity matrix increases, Algorithm 1 correctly identifies clusters in the Spiral dataset. Figure 4 depicts how the evaluation metrics change over time for Algorithm 1. It was found that the criteria NMI and Donn index improve at each stage of Algorithm 1, however, compactness decreases. We believe that the method works by gridding the plane into cells, and if the clusters are present in separate cells, the compactness improves; otherwise, the compactness degenerates. Figure 4 depicts how the Compactness and Donn index change for the Spiral dataset.



(**a**) A (**b**) A^7 (**c**) A^{35}

Figure 3. As we repeatedly apply the transformation discussed in Section 6, the PAM clustering algorithm correctly identifies clusters in the Spiral dataset.

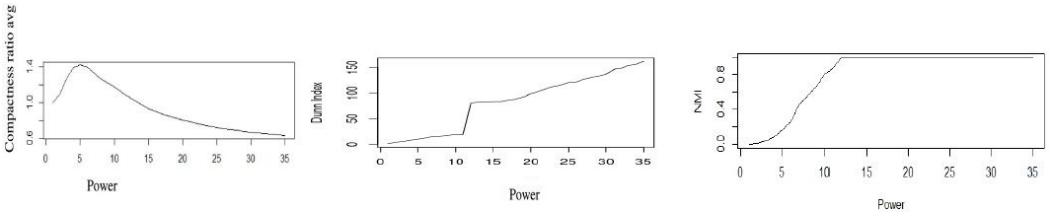


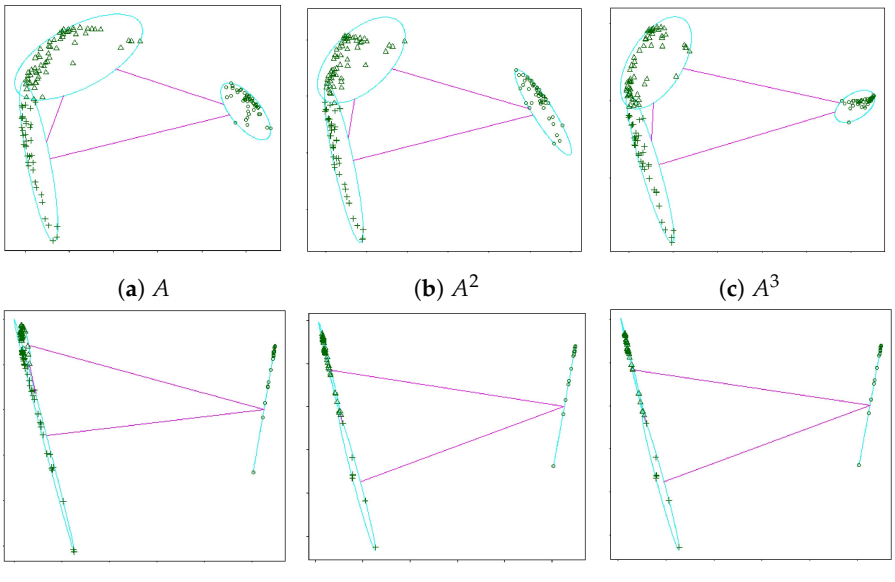
Figure 4. Compactness, Donn index and Normalized mutual information. Spiral dataset.

Table 2 displays the numerical values of compactness and the Donn index at each stage of Algorithm 1. It demonstrates that they improve for the Spiral dataset as the ultrametricity of the dataset's dissimilarity matrix increases.

Table 2. Compactness, Donn index as we increase ultrametricity. Spiral dataset.

	Compactness Avg.	**Compactness Ratio Avg.**	**Dunn Index**	**Dunn Index Ratio**
B^1	0.456225532	1	0.01110	1
B^2	0.499903924	1.095738596	0.042091	3.7914944142
B^7	0.60707534	1.330647448	0.169928	15.3067877
B^{35}	0.286708931	0.628436839	1.7971900	161.88682261

Figure 5 depicts the correct identification of clusters in the Iris dataset. It also shows that as the dataset's ultrametricity increases, so does its compactness.



(**a**) A (**b**) A^2 (**c**) A^3

(**d**) A^7 (**e**) A^{10} (**f**) A^{15}

Figure 5. As the ultrametricity of the distance matrix of the Iris data set increases, clusters are better separated.

Figure 6 shows the improvements in the Iris dataset's sum of squared errors (SSE), Donn index, and compactness. Table 3 also shows that increasing the ultrametricity improves the Donn index at each iteration of Algorithm 1.

Figure 7, shows that increasing ultrametricity results in clusterings with a higher compactness for the Seeds dataset.

Figure 8, shows the improvements in clustering Donn index. It also shows that the sum of squared errors is decreasing for the Seeds dataset as we increase the ultrametricity.

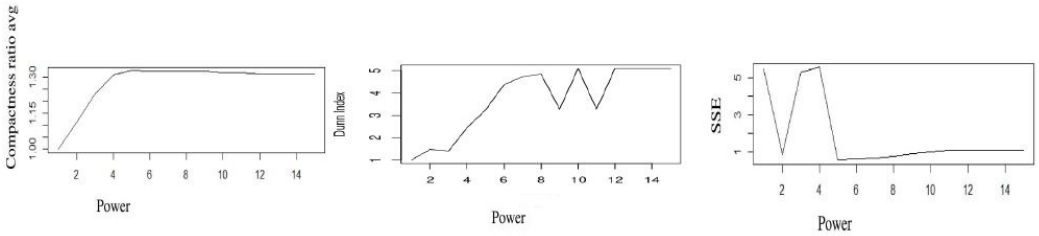
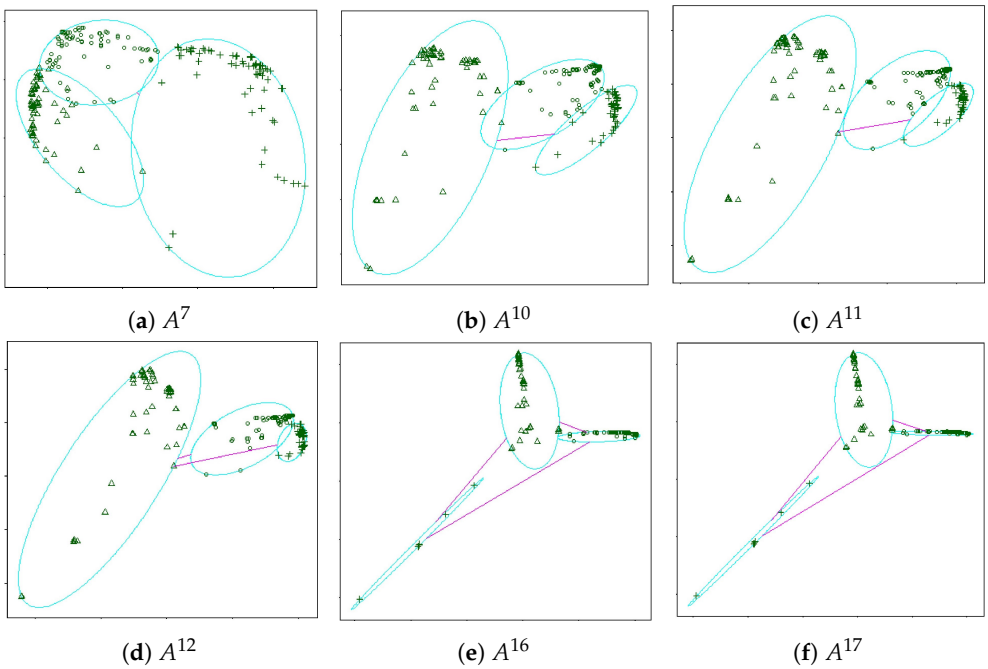


Figure 6. Compactness, Donn index and Sum of Squared errors. Iris dataset.

Table 3. Compactness, Donn index as we increase ultrametricity. Iris dataset.

	Compactness Avg.	Compactness Ratio Avg.	Dunn Index	Dunn Index Ratio
B^1	0.258152	1	0.09880	1
B^2	0.287839	1.114996	0.14560	1.47364530
B^3	0.317521	1.229975	0.138675	1.4034885859
B^7	0.34202	1.324878	0.46684	4.7248466



(**a**) A^7 (**b**) A^{10} (**c**) A^{11}

(**d**) A^{12} (**e**) A^{16} (**f**) A^{17}

Figure 7. As the ultrametriciy of the Seed dataset increases, clusters become more distinct.

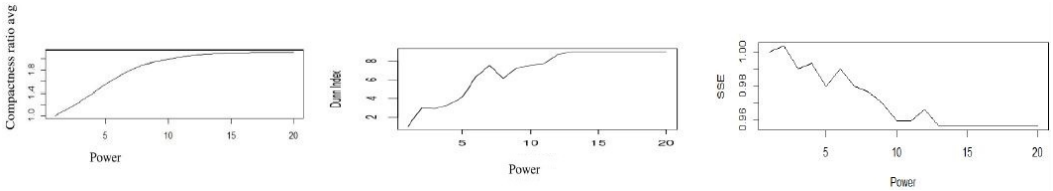


Figure 8. Compactness, Donn index and Sum of Squared errors. Seed dataset.

Table 4 contains the exact value of Compactness and the Donn index.

Table 4. Compactness, Donn index as we increase ultrametricity. Seeds dataset.

	Compactness Avg.	Compactness Ratio Avg.	Dunn Index	Dunn Index Ratio
B^1	0.339115	1	0.049510	1
B^6	0.570942	1.683624	0.312755	6.31701349
B^{10}	0.67483	1.989974	0.37255	7.52478121
B^{12}	0.70014	2.06461	0.429812	8.681313
B^{20}	0.716623	2.113215	0.445516	8.9985077

The ultrametricity of data sets Iris, Seeds, Spiral, Jain, Yeast, and Breast cancer was then assessed using Rammal's measure, as discussed in [13]. In Rammal's measure, smaller numbers represent higher ultrametricity. Using the measure defined in Section 4, we also calculated the degree of ultrametricity for the aforementioned datasets. Figure 9 on the following page compares the proposed measure to Rammal's measure. We see that the two measures are pointing in the same direction and are linearly correlated with a Pearson correlation of 0.9528. For a better demonstration of the relationship, a blue regression line of degree one is also plotted.

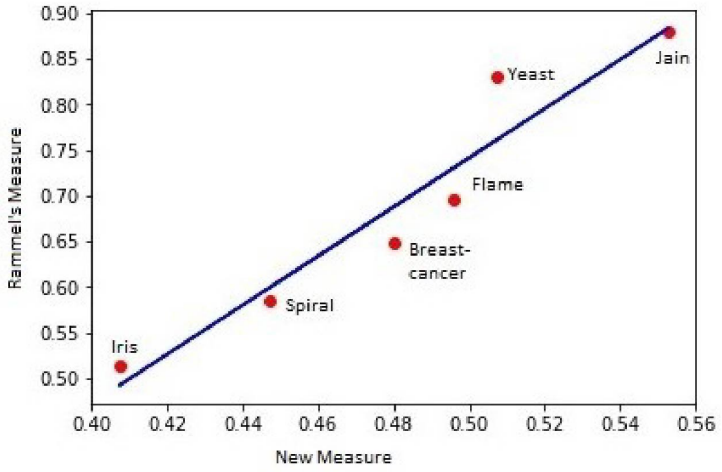


Figure 9. Rammal's measure is pointing in the same direction as our new measure.

To have a preview of the datasets used in the Ultrametric-FABMAP Figure 10 is plotted.

Table 5 displays the recall and accuracy results of the Ultrametric-FABMAP, Ultrametric-BoW, regular FABMAP2, and BoW algorithms.

Newer College one loop is considered as the training data, while Newer College three loop is considered as the test data. Table 6 shows the results of applying Ultrametric-BoW on Lip6indoor dataset. We also switched the training and testing sets to see if there was any improvement in accuracy and recall. Table 7 displays the results of using Ultrametric-FABMAP and the remaining methods in Table 5. In this case, the Newer College three loop is considered training data, while the Newer College one loop is considered test data. Figure 11 depicts the rationale for using ultrametric-BoW and ultrametric-FABMAP as our ground truth.

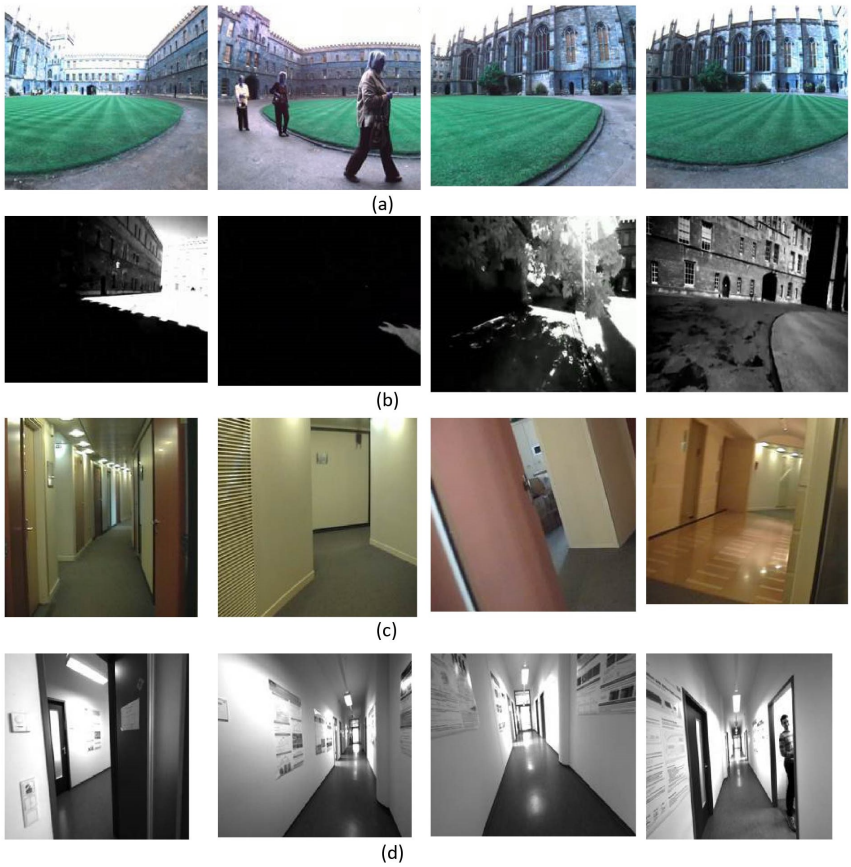


Figure 10. Example of datasets used in Ultrametric-FABMAP.

Table 5. Comparison of FABMAP2, Ultrametric-FABMAP and BOW considering Ultrametric-FABMAP as our ground truth.

Dataset	**No, Extracted Features**	**Method**	**BoW**	
Train/Test	**Train/Test**	————	**%Acc**	**%Rec**
Newer College One Loop (Train)	11,208	Ultrametric FABMAP	Ground Truth	Ground Truth
		Ultrametric BoW	%89.6	%97.4
Newer College Three Loops (Test)	6736	FABMAP2	%50.3	%100
			%54.83	%90.9
			%59.55	%81.5
		BoW	%42	%93
			%96	%86

Table 6. Accuracy and Recall for Lip6indoor dataset considerin TUM sequence 16 as training data.

Dataset	**Method**	**Acc**	**Recall**
Lip6 Indoor	Ultrametric	%68.11	%66
	Bow	%57	%87

Table 7. Comparison of FABMAP2, Ultrametric-FABMAP and BOW considering Ultrametric-BoW as our ground truth (Training and testing data are swapped according to Table 5).

Dataset	No, Extracted Features	Method	Acc	Recall
Train/Test	**Train/Test**	_____	___	_____
Newer College One Loop (Train)	11,208	Ultrametric FABMAP	%57	%93
			%62	%84
			%66.1	%81.66
		Ultrametric BoW	Ground Truth	Ground Truth
Newer College Three loops (Test)	6736	FABMAP2	%51	%92.9
			%61	%83
		BoW	%63	%100

We run experiments on real-world-size datasets Lip6indoor (publicly available at: https//animatlab.lip6.fr/AngeliVideosEn, accessed on 2 March 2019) and Newer College (publicly available at: https://ori.ox.ac.uk/publications/datasets/, accessed on 20 March 2022), which is a video (which we converted into 200 image sequences), and sequence 16 Monocular Visual Odometry collected at Technical University of Munich (TUM) (publicly available at: https://vision.in.tum.de/data/datasets/mono-dataset, accessed on 20 March 2020)). Sequence 16 of TUM is simple loop within a corridor. In this case, we use Algorithm 4 on a set of specific train/test datasets. To build a Chow-Liu tree in the training phase, we use the clustering centeroids and BoW representation of training images. Then, for each image in the test dataset, we select the corresponding BoW representation and calculate the new place likelihood. Using a motion model, we can either insert a new location or declare a loop closure detection. The confusion matrix is returned by this algorithm. Each (i,j) element in this matrix represents a binary value indicating whether or not images i and j are similar. We define several thresholds and calculate the accuracy and recall of the loop closure as follows:

$$accuracy = \frac{\sum_i \sum_j ((Confusion\ Matrix[i][j] > threshold) \wedge ground\ truth[i][j] == 1)}{\sum_i \sum_j (Confusion\ Matrix[i][j] == 1 > threshold)} \quad (12)$$

$$recall = \frac{\sum_i \sum_j ((Confusion\ Matrix[i][j] > threshold) \wedge ground\ truth[i][j] == 1)}{\sum_i \sum_j ground\ truth[i][j] == 1} \quad (13)$$

Figure 11's rows and columns both represent images, and the value of the heatmap for parts (a) and (b) is the cosine similarity between the image's TF-IDF representations. For parts (c) and (d), it is a binary confusion matrix value returned by the FABMAP algorithm, where 1 indicates that two images are similar and zero indicates that they are not. The brighter the points in the confusion matrix, the more similar the corresponding images are.

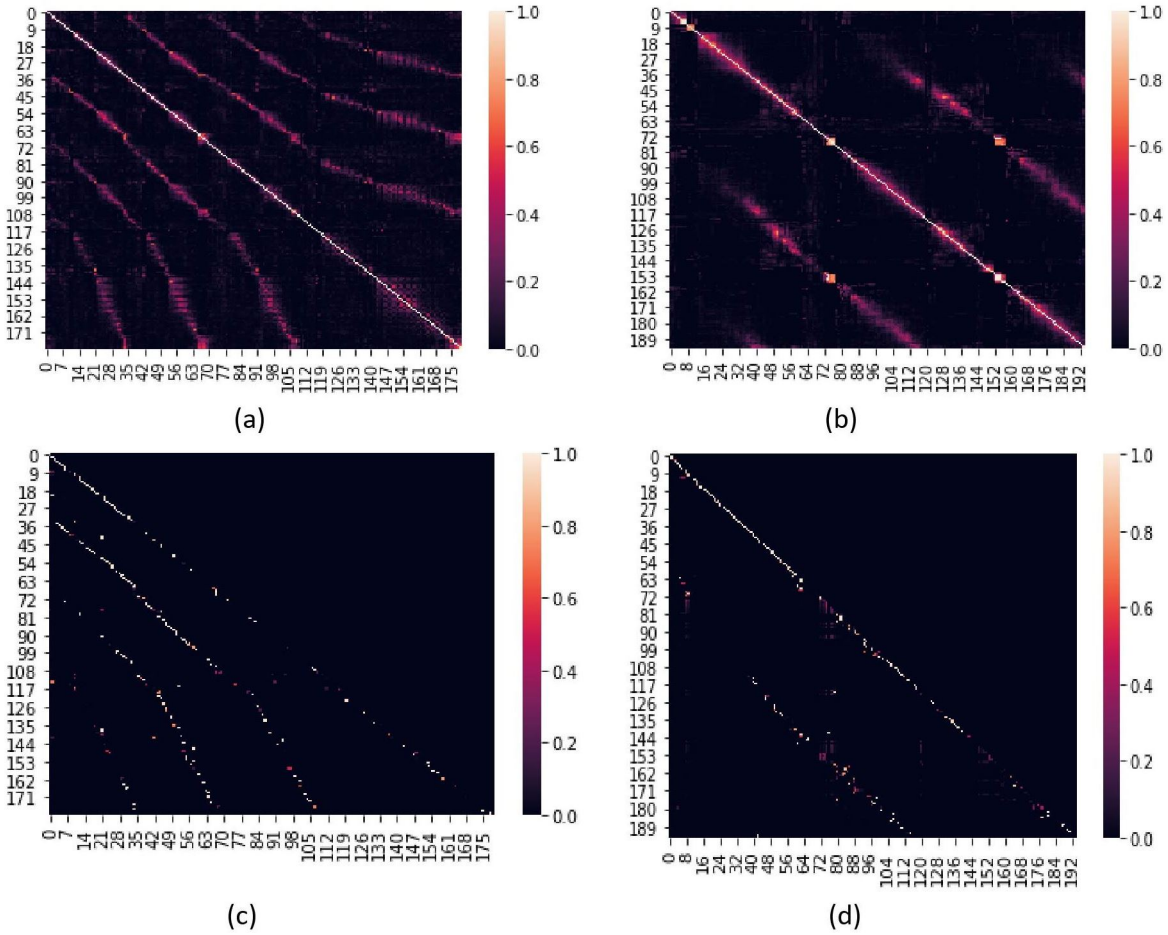


Figure 11. (**a**) Newer College dataset (three loops), (**b**) Portion of New College dataset (one loop) (Ultrametric-BoW), (**c**) Newer College dataset (three loops), (**d**) Portion of New College dataset) (one loop (Ultrametric-FABMAP).

To determine whether we needed to use OpenMP/AVX-256 or a GPU implementation with the following specifications, we compared the speeds of Algorithms 2 and 3. Table 8 shows that using the CUDA C implementation as our base method, OpenMp/AVX-56 improves transformation speed by 50% over CUDA implementation. Experiments are carried out on a Lenovo ThinkCenter i5, 8500, which has a total of six 3.00 GHz cores and six threads. To achieve the greatest possible speedup, we chose 20 OpenMp threads. In contrast, we used a Geforce 710 GT GPU with 192 CUDA cores and a total of 2GB of memory. It is also worth noting that this is the cost of one iteration of the algorithm, and we have set the algorithm to not exceed 500 multiplications. We compared the speed of clustering in both cases to demonstrate that Ultrametric-PAM is superior for clustering, not only in terms of clustering compactness, NMI, SSE, and Donn index, but also in terms of speed.

Table 8. OpenMP/AVX-256 Implementation vs. GPU speed up.

Dataset-Size	Method	Time	Speedup
6736 × 6736	AVX-256	1 min 6 s	1.51
6736 × 6736	GPU	1 min 39 s	1.

Table 9 shows that if we use a comparatively hierarchical distance matrix, we can speed up the PAM clustering algorithm up to 1254 times faster.

Table 9. PAM vs. Ultrametric-PAM speed of clustering.

Dataset-Size	PAM	Ultrametric-PAM	Speedup
6736 × 6736	3.5 days	18 min	×280
11,712 × 11,712	14 days	2 h and 23 min	×1254.

7. Conclusions

In this paper, we demonstrated that the degree of ultrametricity has a direct relationship with the performance of the PAM clustering algorithm. Ultrametric PAM is a thousand times faster than regular PAM. The performance was also evaluated using the clustering Donn index, compactness, sum of squared errors, and normalised mutual information. Clusters with a better Donn index, higher normalised mutual information, and a lower sum of squared errors are obtained by using the PAM algorithm in spaces with higher ultrametric configuration. Furthermore, we demonstrated that our new measure and Rammal's measure are pointing in the same direction, with a Pearson's correlation close to 1. It has been demonstrated that adding a hierarchical structure to the dataset distance matrices improves the accuracy and recall of loop closure detection by at least 6%. In future works, we will attempt to define Boolean Strassen matrix multiplication in order to reduce transformation time calculation.

Author Contributions: Conceptualization, S.S.G., A.Z., A.M. and S.-A.S.-Z.; methodology, S.S.G., A.Z., A.M. and S.-A.S.-Z.; software, A.Z.; validation, A.Z.; formal analysis, S.S.G., A.Z., A.M. and S.-A.S.-Z.; investigation, A.Z., S.S.G., A.M. and S.-A.S.-Z.; resources, A.Z., S.S.G. and S.-A.S.-Z.; writing—original draft preparation, S.S.G., A.Z.; writing—review and editing, S.S.G., A.Z. and A.M.; visualization, A.Z.; supervision, S.S.G., A.Z. and A.M.; project administration, S.S.G., A.Z. and A.M.; funding acquisition, S.-A.S.-Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study such as Jain, Seed, Iris, Spiral. This data can be found here: https://archive.ics.uci.edu/ml/index.php (accessed on 20 December 2017). Also dataset Lip6indoor is publicly available in: https//animatlab.lip6.fr/AngeliVideosEn (accessed on 2 March 2019), Dataset Newer College is a video (which we have converted into 200 image sequence), which is publicly available here: https://ori.ox.ac.uk/publications/datasets/ (accessed on 15 May 2022) and dataset TUM sequence 16 is publicly available at: https://vision.in.tum.de/data/datasets/mono-dataset (accessed on1 May 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Simovici, D.; Hua, K. Data Ultrametricity and Clusterability. *J. Phys. Conf. Ser.* **2020**, *1334*, 012002. [CrossRef]
2. Tan, Y. On the powers of matrices over a distributive lattice. *Linear Algebra Its Appl.* **2001**, *336*, 1–14. [CrossRef]
3. Tan, Y.J. On the transitive matrices over distributive lattices. *Linear Algebra Its Appl.* **2005**, *400*, 169–191. [CrossRef]
4. Murtagh, F. Sparse p-adic data coding for computationally efficient and effective big data analytics. *P-Adic Numbers Ultrametric Anal. Appl.* **2016**, *8*, 27–42. [CrossRef]
5. Bradley, P.E.; Keller, S.; Weinmann, M. Unsupervised Feature Selection Based on Ultrametricity and Sparse Training Data: A Case Study for the Classification of High-Dimensional Hyperspectral Data. *Remote Sens.* **2018**, *10*, 1564. [CrossRef]
6. Murtagh, F. Identifying and Exploiting Ultrametricity. In *Proceedings of the Advances in Data Analysis, Proceedings of the 30th Annual Conference of the Gesellschaft für Klassifikation e.V., Freie Universität Berlin, 8–10 March 2006*; Decker, R., Lenz, H., Eds.; Springer: New York, NY, USA, 2006; pp. 263–272. [CrossRef]
7. Murtagh, F.; Contreras, P. The Future of Search and Discovery in Big Data Analytics: Ultrametric Information Spaces. *arXiv* **2012**, arXiv:1202.3451. [CrossRef]
8. Kaufman, L.; Rousseeuw, P. *Finding Groups in Data: An Introduction to Cluster Analysis*; John Wiley: New York, NY, USA, 2009.
9. K-Medoids. Available online: https://en.wikipedia.org/wiki/K-medoids (accessed on 10 December 2022).

10. K-Medoids Clustering with Solved Example. Available online: https://www.geeksforgeeks.org/ml-k-medoids-clustering-with-example/ (accessed on 10 December 2022).
11. Cummins, M.J.; Newman, P. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *Int. J. Robot. Res.* **2008**, *27*, 647–665. [CrossRef]
12. Glover, A.; Maddern, W.; Warren, M.; Reid, S.; Milford, M.; Wyeth, G. OpenFABMAP: An Open Source Toolbox for Appearance-based Loop Closure Detection. In Proceedings of the International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; IEEE: St. Paul, MN, USA, 2012.
13. Rammal, R.; Anglès D'Auriac, J.C.; Douçot, B. On the degree of ultrametricity. *J. Phys. Lett.* **1985**, *46*, 945–952. [CrossRef]
14. Cattaneo, D.; Vaghi, M.; Valada, A. LCDNet: Deep Loop Closure Detection for LiDAR SLAM based on Unbalanced Optimal Transport. *arXiv* **2021**, arXiv:2103.05056.
15. Lu, S.; Xu, X.; Tang, L.; Xiong, R.; Wang, Y. DeepRING: Learning Roto-translation Invariant Representation for LiDAR based Place Recognition. *arXiv* **2022**, arXiv:2210.11029.
16. Zarringhalam, A.; Ghidary, S.S.; Khorasani, A.M. Self-supervised Vector-Quantization in Visual SLAM using Deep Convolutional Autoencoders. *arXiv* **2022**, arXiv:2207.06732.
17. Zarringhalam, A.; Ghidary, S.S.; Khorasani, A.M. Semi-supervised Vector-Quantization in Visual SLAM using HGCN. *arXiv* **2022**, arXiv:2207.06738.
18. Murtagh, F.; Downs, G.; Contreras, P. Hierarchical Clustering of Massive, High Dimensional Data Sets by Exploiting Ultrametric Embedding. *SIAM J. Sci. Comput.* **2008**, *30*, 707–730. [CrossRef]
19. Gillet, V.J.; Wild, D.J.; Willett, P.; Bradshaw, J. Similarity and Dissimilarity Methods for Processing Chemical Structure Databases. *Comput. J.* **1998**, *41*, 547–558. [CrossRef]
20. Brown, R.D.; Martin, Y.C. Use of Structure–Activity Data To Compare Structure-Based Clustering Methods and Descriptors for Use in Compound Selection. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 572–584. [CrossRef]
21. Downs, G.M.; Willett, P.; Fisanick, W. Similarity Searching and Clustering of Chemical-Structure Databases Using Molecular Property Data. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 1094–1102. [CrossRef]
22. Rammal, R.; Anglès D'Auriac, J.C.; Douçot, B. Several Remarks on Dissimilarities and Ultrametrics. *Sci. Ann. Comput. Sci.* **2015**, *25*, 155–170.