Application and evaluation of a K-Medoids-based shape clustering method for an articulated design space

Shermeen Yousif 1,* and Wei Yan 2

1School of Architecture, Florida Atlantic University, 111 E. Las Olas Blvd. Ft Lauderdale FL 33301, USA and 2Department of Architecture, Texas A&M University, 3137 TAMU College Station, TX 77843, USA

*Corresponding author. E-mail: syousif@fau.edu

Abstract

Research on articulating the design space in computational generative systems is ongoing, to overcome the issue of possible overwhelming multiplicity and redundancy of emerging design options. The article contributes to this line of research of design space articulation, in order to facilitate designers’ successful exploration in computational design. We have recently developed a method for shape clustering using K-Medoids, a machine learning-based strategy. The method performs clustering of similar design shapes and retrieves a representative shape for each cluster in 2D grid-based representation. In this paper, we present a progress in our project where the method has been applied to a new test case, and empirically verified using clustering evaluation methods. Our clustering evaluation results show comparable accuracy when assessed against an external study and provide insight into the evaluation criteria for machine learning methods, as presented in the paper.

Keywords: design space articulation; generative design systems; shape clustering; clustering evaluation methods

1. Introduction

Generative design systems (GDSs) are computational frameworks that involve parametric modeling and form finding, the search for the most successful or best candidates in a generated set of designs, often accompanied by a process of evaluation of form quality and/or performance (Barnes, 1999). Despite their significance as promising schemes, one drawback in available GDSs is the inherent possibility of producing an excessive number of designs, thousands and more, making it difficult for designers to cope with such systems (Rodrigues et al., 2017). Also, the produced set of designs may have redundant and similar characteristics, adding computational burden and inhibiting effective design exploration. The creative process of generating and exploring the design space is performed by designers, and thus, the design space should be supportive of successful navigation and interaction between the designers and the machine (Brown & Mueller, 2019).

Existing generative systems lack established organizational mechanisms for successful form finding (Rodrigues et al., 2017; Brown & Mueller, 2019). Alternatively, it would be easier for designers to navigate through an articulated design space where similar designs are grouped into subsets yet different from the other subsets, and representative designs of each group are highlighted (Yousif & Yan, 2019a). This way, examining the organized clusters becomes feasible and designers can focus on analysing particular designs of interest. This approach can be achieved through clustering mechanisms, which are machine learning (ML)-based strategies. For a set of data, clustering is associated with partitioning and finding hidden patterns in an unsupervised manner (Velmurugan & Santhanam,
where the purpose is exploratory. However, for a dataset of architectural designs, applying clustering requires other tasks such as establishing methods for shape comparison and similarity/difference finding, which was the motivation of this research. The basic concept of this study is that organizational methods can be incorporated into GDSs for an articulated design space. Such an organized system supports successful human–computer interaction and facilitates an effective search for preferred design options, within a multiplicity of designs.

This research introduces a novel GDS with the introduction of an architectural shape clustering mechanism integrated into the search process. In previous published works, the focus was on developing a shape clustering method to compare the geometric characteristics of design shapes in 2D representation and find correspondences for identifying similarities and differences (Yousif & Yan, 2019a, b) that led to the development of a new shape difference finding method. In addition, prior work involved the implementation of the K-Medoids clustering mechanism into a method we called shape clustering using K-Medoids (SC-KM). In this paper, a new experimental test case was carried out, aimed at (1) applying the SC-KM method to a new dataset of shapes, (2) further developing the method, adding an algorithm to convert the boundary-based building design shapes into grid-based shapes for shape comparisons, and (3) utilizing clustering evaluation metrics to test the clustering results in comparison with an existing study as an external validation procedure. Clustering comparison measures are often used to assess the performance or “goodness” of the clustering procedures (Vinh et al., 2010). Importantly, evaluating ML strategies, in particular unsupervised methods such as clustering, requires a thorough investigation and therefore has been also targeted in this study. Such an assessment of the results of ML is important for advancing the integration of artificial intelligence methods into computational design processes.

The paper is structured as follows: In Section 2, related literature is reviewed, and definitions and explanations of important technical concepts that are most relevant to this work are provided. Section 3 is dedicated to describing the overall methods and strategies of our developed system. In Section 4, a detailed description of the testing and validation procedures are presented. At the end of the paper, discussions of the experimental results are presented in Section 5, and overall conclusions and future work propositions are outlined in Section 6.

## 2. Background

Articulating design solutions produced in computational generative schemes has only recently been addressed (Rodrigues et al., 2017; Brown & Mueller, 2019). Effective exploration of the design problem requires a portrayal of corresponding geometry of designs under analysis (Turrin et al., 2016). The study of Brown and Mueller reviews different diversity metrics used in generative design protocols for design space articulation (2019). Seeking design diversity among the possible design alternatives is important to avoid obtaining, or simulating, repeated and similar candidates and to enhance generative mechanisms to ensure that “the results they produce are diverse enough to be interesting to designers” (Rodrigues et al., 2017, p. 2). In a prior work within this research project, a diversity measure was developed to condense the design set into a highly diverse one (Yousif et al., 2017). In continuing experimentation with design diversity, it became obvious that clustering methods have the capability to lead not only to a highly diverse set of designs, but also retain and organize all designs. Therefore, clustering techniques were investigated.

In organizing big data, two methods are typically pursued: classification and clustering. Clustering is labeled as an unsupervised learning method, as it involves the discovery of a structure or organizing a collection of unlabeled data (Jain et al., 1999; Velmurugan & Santhanan, 2010). Unlike classification, which deals with predefined classes, clustering does not tackle classified data, making it advantageous to find interesting hidden patterns with no predefined knowledge (Han et al., 2011). The integration of clustering methods into GDSs is still experimental and limited. One of the few related studies is the work of Rodrigues et al. that compares multiple descriptors of 2D shapes and utilizes the Ward linkage clustering method for architectural floor plans (Ward Jr, 1963; Rodrigues et al., 2017). The authors point out that the need to further investigate clustering algorithms for architectural layouts. Another relevant study is the work of Harding and Brandt-Olsen (2018) on combining parametric modeling with an interactive Cluster-Oriented Genetic Algorithm, introducing interactive evolution and combining modeling and analysis into the generative process. It uses “K-Means+” clustering to visualize similar design alternatives; however, the study is focused on the evolutionary architecture approach. More recently, cluster analysis has been integrated into urban morphological analysis (Cai & Li, 2020), and coupled with deep learning strategies (Li, 2020).

Multiple clustering tools for mesh segmentation have been investigated. In one example, an attempt to introduce clustering, particularly K-Means clustering, into generative design in visual programming platforms has been done by the authors of the Ivy tool for Grasshopper® (Nejur & Steinfeld, 2016). Yet, the K-Means clustering in the tool was applied to mesh segmentation and not to the entire building form. Other tools that use clustering include the ML-based tools: “Owl” (Zwierzycki et al., 2018) and “Ant” (Abdelrahman & Toutou, 2019), both for the Rhino/Grasshopper® platform. The K-Means clustering in the Owl tool has been tested in experimentation studies for this research. These tools are problem specific and not particularly targeted to achieve shape clustering based on shape difference analysis. In mechanical engineering and product design, the work of Jayanti et al. (2009) represents one of the early attempts to address the need for clustering in managing CAD repositories to sort and retrieve the three-dimensional (3D) models according to shape similarity. The study compares five shape representation techniques that are often applied in engineering, targeted for clustering evaluation. It considers the 2D drawings of the 3D CAD models and applies a K-Means clustering method (Jayanti et al., 2009). The work also focuses on clustering result assessment mechanisms and encourages further investigation of how different clustering algorithms are suitable for different shape representations (Jayanti et al., 2009).

Shape representation and shape comparison are significant for performing the shape clustering method investigated here. Shape representation is associated with finding effective and descriptive shape features (Zhang & Lu, 2004). An area related to shape comparison is pattern recognition. The work of Cha and Gero (1998) has laid foundations for a shape pattern recognition system based on structural shape representation. In another work, de las Heras et al. (2013) have used an approach to retrieve designs with similar properties from a dataset, while Dutta et al. (2013) applied a graph-based method to recognize symbols in floor plans such as furniture and fenestration.

Given this background, the lack of established shape clustering methods in GDSs remains an unsolved problem. Therefore,
this study was targeted to improve the recently formulated SC-KM method and test its clustering performance when applied to other cases, as explained in the Methods and the Test-Case Application sections. Before that, it is important to introduce background information on the methods and algorithms utilized in the SC-KM, as described in the following subsections.

2.1 The hamming distance for grid-based shape difference

Prior to comparing shapes and finding geometric correspondences, a method of shape description or representation is needed. In our investigation, the focus was on grid-based shape description. In grid-based shape description, the shape under analysis is mapped with a grid of cells and scanned in left-to-right, top-to-bottom manner resulting in a bitmap (Zhang & Lu, 2004). Using a binary mapping, the cells covered by the shapes (or the center of the cell covered by the shape) are assigned 1, and those not overlapped by the shapes are assigned 0. The shape becomes represented by a matrix or a binary vector of 0s and 1s (Zhang & Lu, 2004). The Hamming distance is typically used to compare binary vectors; the method involves comparing two vectors of equal length and calculating the number of cells that are overlapped by one shape yet not the other (Sajjanhar & Lu, 1997). It is a similarity/difference measure of the minimum number of errors that can convert one binary code to another (Norouzi et al., 2012). In a previous work, we have utilized this Hamming distance method to retrieve shape difference scores; in our calculation, for each pair-wise shape comparison, the shape difference score represents the number of nonoverlapping cells between the two shapes after searching for maximum overlap (Yousif et al., 2017). A cross-reference matrix of the shape difference scores was utilized following the Hamming distance method, and the results yielded that the higher the score, the more different the shapes are.

In developing our method, it became clear that the Hamming distance, despite its reliability, is not necessarily the best method for shape difference. The reason for the possible inaccuracy of the Hamming distance is that the sum of the total number of cells that are not corresponding between two shapes does not offer information on the location of those cells that should also be considered for computing the shape difference. As such, we have carried out another approach, developing a distance-based calculation, discussed in the Methods section. Our distance-based shape comparison method calculates and totals the Euclidean distances between nonoverlapping cells in a pairwise shape difference finding. Assigning the nonoverlapping cells between two shapes created an optimization problem that required another algorithm to solve, the Hungarian algorithm.

2.2 The Hungarian algorithm

As a combinatorial optimization technique, the Hungarian algorithm, developed by Kuhn and later revised by Munkers, is a method for solving assignment problems (Korsah et al., 2007). For a linear assignment problem, also called the maximum weighted bipartite matching, the algorithm can be explained as follows: given a set of workers and a set of jobs, and a rating of how well each worker performs each job, it determines the optimum possible assignment of jobs to workers in a way that the total rating is maximized or minimized (Munkres, 1957; Ayorkor et al., 2007). For each overlap case in our pair-wise shape comparison, the Hungarian algorithm is used to match the geometric components (cells in grid-based shapes) between each pair of compared shapes, in order to find the smallest overall sum of Euclidean cell distance between the two shapes in the pair, for calculating the pair’s shape difference measure.

2.3 K-Medoids clustering

Clustering, or cluster analysis, is an area that encompasses a range of methods for partitioning or separating a dataset into a variable number of subsets or clusters. Members within each cluster share high similarity yet are different from the members of other clusters (Han et al., 2011). Unlike classification, clustering is an unsupervised technique where the data’s identities or labels are unknown prior to the process (Wilks, 2011). Clustering has been established and used for more than 80 years, but since the calculations associated with clustering are often difficult, the invention of computers in the 1950s and later advanced computation led to development in clustering methods (Bailey, 1994). Cluster analysis is part of statistical analysis packages and is extensively used in areas of data mining, ML, computer vision, information retrieval, and other areas.

Clustering can be divided into four subcategories: partitioning, hierarchal, distance measurement, and grid-based methods. It is important to note that the grid-based approach referred to here is a clustering method and must not be confused with the grid-based shape description used in this work. Among partitioning methods, K-Means and K-Medoids are widely used. K-Means is a centroid-based clustering technique. It performs partitioning the dataset into a k number of clusters, initiating the cluster centers or means first. It then refines the means in iteration, minimizing the distance between the mean and the data points within the cluster; when there is no more update in assigning data points to their closest means, the algorithm converges (Wagstaff et al., 2001). The quality of a cluster is measured by the within-cluster variation or the total of squared error between all items and the centroid. The objective function is to make the clusters compact and separate (Han et al., 2011). Simply, the algorithm works as follows. For a dataset D, it initiates a centroid for each cluster arbitrarily, as the mean value of the data point for each cluster. Next, it proceeds with assigning the remaining items (data points) to their closest mean (using Euclidean distance). Iteratively, the algorithm improves the within-cluster variation as it computes a new mean for each cluster that is closest to all cluster data points; further, all data points are reassigned to the new updated means, forming new clusters. When re-assignment is stable, the new updated clusters converged with the similar older formed clusters, the algorithm terminates and outputs final clusters with their means (Han et al., 2011).

One of the disadvantages of the K-Means clustering method is that it does not perform well for clusters with different sizes and/or with nonconvex shapes since it is sensitive to noise and outlier data points. Outliers that are distant from most data can significantly influence the mean value and distort the partitioning (Han et al., 2011). To reduce such sensitivity, instead of using the mean value as a reference point for each cluster, an actual object or data point can be used. The most central object becomes a representative of the cluster. This approach is the partitioning around medoids or K-Medoids clustering. The algorithm assigns data to clusters in a greedy and iterative manner, as all the possible replacements of nonmedoids within the cluster medoid test each other to improve the clustering quality (Han et al., 2011). Applying the K-Medoids clustering method was pursued in this work, since the medoid can be an existing design alternative within a cluster of similar
design shapes. To implement the K-Medoids to a dataset of shapes, there was a need to investigate scientific methods to find differences within datasets of shapes that required developing the pair-wise shape difference finding as explained next.

3. Research Methods

The methodology of this work involved an extensive literature study, experimenting and prototyping, and testing and validation. These methods have led to the development of a new shape clustering method, the (SC-KM). Developing the method, the protocol included employing a grid-based descriptor, formulating a shape difference finding method, and implementing the K-Medoids clustering algorithm. In our previous publications, the SC-KM method was fully described (Yousif & Yan, 2019a, b). Experimenting and prototyping were performed in the Rhino/Grasshopper® environment. For the grid-based shape generation and description, modeling was pursued, using visual programming, in addition to customized programs written in Grasshopper® Python, and C# languages. For the SC-KM, a package of algorithmic set was developed, primarily using the GH, CPython tool (AbdelRahman, 2017) that allows communicating with the Python environment, and incorporating the scientific libraries and modules. For explaining the test-case application and evaluation, the focus of this paper, there is a need to concisely describe the SC-KM method and introduce its algorithms.

For shape description, a typical grid-based approach was employed to define the shape characteristics. The shape difference finding method we developed started with investigating the distance-based diversity measure (Toffolo & Benini, 2003). Since the dataset in our case is a set of shapes, we needed a method applied to multidimensional space such as the architectural design space. As such, we formulated two sets of algorithms: (1) pair-wise shape difference and the Hungarian algorithm, and (2) K-Medoids clustering, depicted as a flowchart in Fig. 1.

3.1 Pair-wise shape difference and the Hungarian algorithm

For the shape difference method, a distance-based calculation for finding shape difference/similarity was needed. The method computes the distances between the center points of the nonoverlapping cells in one shape and the center points of the nonoverlapping cells of the other shape, in a process of pair-wise shape difference finding. The total of Euclidean distances between the nonoverlapping cells of the compared pair becomes the shape difference score. To find optimum assignment of the nonoverlapping cells between the two shapes, the Hungarian algorithm was utilized. For each pair of shapes compared, a search mechanism was utilized, implementing the Hungarian algorithm to find the best assignment in terms of minimal total Euclidean distance of the nonoverlapping cells among all overlapping cases, and this smallest total score represents the definite shape difference. The result of the cross-reference pair-wise shape comparison is a matrix of shape difference scores.

3.2 K-Medoids clustering

In implementing the K-Medoids clustering, the matrix of cross-reference shape difference scores retrieved from the developed shape comparison approach has become the input for clustering. This clustering algorithm outputs a variable number of subsets; in each subset, shapes are similar yet are variant from the shapes in other subsets. We applied the K-Medoids clustering to a sample of 2D shapes that emerged from parametric generative systems and the method clustered the dataset into groups of similar shapes, and identified a medoid, a representative shape for each cluster, as demonstrated in Yousif and Yan (2019b). In multiple test cases, the SC-KM method showed successful clustering results, when evaluated according to perceptual coherence—visual examination of each cluster’s coherence. Figure 2 depicts one of those cases, in which 100 shapes of architectural typological designs of 48-cell grid-based representation were clustered into 10 clusters, and 10 representative shapes are highlighted (circled).

As far as our experimentation workflow is concerned, the SC-KM method is incorporated after shape description in generative systems, inserted post to modeling and design evaluation (including performance simulation and/or optimization). However, the objective of incorporating the SC-KM method is that it can be implemented immediately after modeling and initial parametric design generation. This way, it enables excluding similar and unwanted designs and maintaining diverse options, which saves computation in the case of performance evaluation/design optimization.

4. Test-Case Application and Evaluation

In order to evaluate the developed SC-KM method, this experimental test case was carried out. In comparison with another shape clustering study, our method was applied to a set of 72 shapes from the study of Rodrigues et al. (2017). The shapes represent designs of architectural floor plans for a three-bedroom single-family house, which evolved in a hybrid computation.
framework that combines an evolutionary program for space allocation with a Stochastic Hill Climbing method (Rodrigues et al., 2013). The study applied the Ward Linkage clustering to those 72 shapes, resulting in the outcome that we compare our results against, as explained in Section 4.3. For creating a reference set, the 72 shapes were grouped manually according to their typological characteristics, i.e. L-Shapes, T-Shapes, rectangles, etc. (Rodrigues et al., 2017) as depicted in Fig. 5. It is important to note that the method used in synthesizing and clustering the set of designs in Rodrigues et al.’s work and in our work is invariant to scale, yet variant to rotating and reflecting procedures.

For testing and comparison purposes, we used the same reference set and the clustering evaluation metrics used in Rodrigues et al.’s work, yet we implemented additional metrics. Our framework involved (1) modeling the synthetic dataset of shapes and applying a packing algorithm for grid-based description, (2) applying the SC-KM method, and (3) evaluating the clustering outcome in a quantifiable manner using clustering assessment metrics. Those tasks are described in the following subsections and diagrammed in Fig. 3.

### 4.1 Modeling and packing algorithm

The shapes were modeled as boundary based using the visual algorithmic nodes in Grasshopper®. In addition, a new algorithm was required to convert the boundary-based shapes into...
Application and evaluation of the SC-KM method

4.1 Grid-based Packing

The grid-based packing method applied to the nine typological shapes of the reference set. Left: 36-cell packing. Right: 64-cell packing.

Our packing algorithm is based on the bin-packing approach that seeks to pack a set of different size items into a minimum number of identical bins (Korf, 2002). Yet, in our packing technique, the items to be packed are unified in size, and the bins (the shapes to be packed) vary in size and characteristics. Thus, we formulated a packing method, developing a new Python program and using the Grasshopper plugins. This involved three operations, described as follows:

i. Scaling and translation: Each of the 72 shapes was scaled to match the area of all other shapes, since originally the shapes were of different sizes. Also, each shape was surrounded by a bounding box and translated (moved) so that the upper left corner is at the origin point (x = 0, y = 0).

ii. Packing with an array of cells: A simple Python code was written to create an array of units (each unit is 1 x 1). Next, a checking algorithm was followed to separate the cells' center points that are inside the shapes and those outside the shape. Importantly, the number of packed area units (number of cells) could be parametrically changed to allow a range of number of units to be packed.

iii. Normalizing the list of contained cells’ center points: The list of center points that are contained in the shape has been retrieved and subjected to analysis to check their numbers. All shapes had to be packed with the same number of cells, which was not always the case. This procedure is required for an accurate grid-based shape description.

For comparison, the packing procedure was applied to two cases: Scenario 1 of 36-cell packing and Scenario 2 with 64-cell packing. For illustrating the packing process, the nine representative shapes of the reference set are demonstrated in Fig. 4, showing the 36-cell packing on the left-hand side and the 64-cell packing scenario on the right-hand side. The packing size parameter was primarily considered for testing the expected performance of the shape difference calculation; i.e. higher resolution leads to higher accuracy in the calculation. The decision to increase to 64 was an attempt to approximate the number of cells packed in the grid-based representation in Rodrigues et al.’s work. In future work, we aim to test more packing sizes, and we aim to create a graph to compare which grid resolution will be optimal for accuracy and computation.

4.2 Applying the SC-KM method

After packing, calculation procedures of the “Pairwise Shape Difference and the Hungarian Algorithm” for the two experiments were conducted. Computation wise, particularly for the 64-cell packing, this task was heavy. The shape difference calculation has led to (72 x 71 / 2 = 2556) calculation steps. Thus, the calculation was performed in an agglomerative manner where in each step, one case of pair-wise comparison is calculated for finding the pair’s shape difference value within a batch-run approach. This was performed using an Intel® Core™ i7 8086K (6-Core/12-Thread, 12MB Cache) processor, and a video card of Dual NVIDIA® GeForce® GTX 1080 Ti graphics with 11GB GDDR5X each. For Scenario 1, in each step of the batch run, the two compared shapes were overlapped in (36 x 36 = 1296) overlap cases for every single pair-wise comparison, of which the computation time needed was 10 seconds. Therefore, the total cases of 2556 comparisons required 25560 seconds, or 7 hours and 6 minutes. While for Scenario 2, every pair-wise shape difference calculation took 80 seconds, leading to an overall calculation of the “Shape Difference Score Matrix” of (80 x 2556 = 204480) seconds, or 56 hours and 48 minutes.

In order to improve the performance of the current algorithm, in future work we will investigate the use of integrated Hamming Distance and Hungarian algorithm to significantly reduce computation. For example, the approach will follow the tasks of (i) first, using the Hamming Distance algorithm (more computationally efficient than the Hungarian algorithm), (ii) sorting the Hamming Distances, and then (iii) using the Hungarian algorithm to find the actual shape distance. The threshold will need to be found in the study. In addition, parallel computing and ML methods will be experimented with to improve the
performance of the current algorithm. A generative design process would ultimately require real-time or near-real-time shape clustering performance, which is not accomplished in this work. However, in addition to the suggested changes in future work, another major application of the current algorithm is to synthesize training data for ML toward improved performance, which is a future research direction to advance the topic of shape clustering.

Implementing the K-Medoids clustering algorithm, the “Shape Difference Score Matrix” in both scenarios becomes the main input. In addition, the variable of “Number of Clusters” was kept 9, for the purpose of comparison to the reference clustering and the grid-based clustering result of Rodrigues et al. (2017). When it comes to computation time, the K-Medoids clustering algorithm executes the program in less than 5 seconds, the algorithm organizes the set and performs the clustering results successfully. The outputs of the K-Medoids program were the nested list of clusters with their shapes’ IDs and the IDs of the medoids. Articulating the outputs and visualizing them were done next, using a color coding to each cluster and a darker tone for the medoid of each cluster.

To visualize and discuss the clustering results, it is important to note that in our algorithmic clustering, the shape type or label is unknown prior to performing clustering, as it is unsupervised. This means that the reference clustering set was not used to label or classify the dataset; instead, the reference was only used for the result comparison post to performing the clustering algorithmic definition. Despite that the reference clustering of Rodrigues et al. does not necessarily represent the most accurate clustering data or ground truth, as it has been performed subjectively and some shapes can be re-clustered differently, comparing our results to the reference set was pursued for implementing and testing clustering evaluation measures. The outcomes of the clustering method are illustrated in Figs 5 and 6, both in

Figure 5: Above: the reference set; below: the clustering results of the 72 shapes using 36-cell packing, and the medoid of each cluster represented in a darker tone.
relation to the reference set referred to above. In both cases, the clusters were illustrated and color coded according to the same color and label used for the reference clustering.

For Scenario 1 (Fig. 5), the number of shapes per cluster varies from 3 for Cluster B to 16 for Cluster I. The cluster with the highest number of dominant shapes was Cluster I with 11 mirrored L-shapes. Importantly, the clustering results show seven unique dominant groups, represented in the medoids of the seven clusters (A, B, D, E, G, H, and I), and two clusters are repeated (Cluster C can be considered similar to Cluster B despite the different proportions, and Cluster F is similar to Cluster E with some differences). In terms of perceptual coherence of the clusterings, almost every cluster has a dominant shape represented by the medoid (darker tone); however, outliers do exist, when considering the reference clustering. The outliers can be identified as the shapes with different labels and colors from the dominant label and shape. It is noticeable that in a number of cases, two or more typological dominant shapes are present, with the exception of Cluster H, which is with one dominant shape (the mirrored Z-Shape) and without outliers.

In Scenario 2 (Fig. 6), the number of shapes per cluster varies from 5 for clusters F and I to 14 for Cluster D. Overall, the nine clustering sets show seven unique dominant shapes, signified by the seven medoids of the clusters (A, C, D, E, F, G, H), while Cluster B can be considered similar to Cluster A with a dominant rectangular shape, and Cluster I is related to Cluster H with a dominant mirrored Z-Shape. It is noticeable in the resulting sets that overall, perceptual coherence has been relatively improved from the results of the first scenario with a slightly higher number of dominant shapes and somewhat higher accuracy measure, as will be explained in the clustering evaluation subsection. Dominant shapes were completely attained in clusters F and H with no outliers. Other clusters are perceptually coherent, yet include outliers that belong to other dominant shapes, particularly in clusters (A and B) with three or more typified outlier shapes within each cluster, while the remaining sets have two or less typified outliers. For each scenario, the clustering result
Table 1-a: Confusion Matrix 1-a, results of clustering using a grid-based descriptor (Rodrigues et al.) compared to the reference set.

| Reference clustering | A | B | C | D | E | F | G | H | I |
|----------------------|---|---|---|---|---|---|---|---|---|
| A’                   | 3 |   |   |   |   |   |   |   |   |
| B’                   | 1 |   | 3 |   |   | 8 |   | 1 |   |
| C’                   | 2 |   | 3 | 2 |   |   |   |   |   |
| D’                   |   |   |   |   | 1 |   | 1 |   |   |
| E’                   | 1 | 1 | 1 |   |   |   |   |   |   |
| F’                   | 1 |   |   | 1 | 1 |   |   |   |   |
| G’                   | 1 |   | 4 |   |   |   | 3 |   |   |
| H’                   | 2 | 8 | 5 |   |   |   | 3 | 2 |   |

Table 1-b: Confusion Matrix 1-b, results of clustering using 36-cell packing compared to the reference set.

| Reference clustering | A | B | C | D | E | F | G | H | I |
|----------------------|---|---|---|---|---|---|---|---|---|
| A’                   | 2 |   | 5 |   |   |   |   |   |   |
| B’                   | 1 | 1 | 1 |   |   | 2 |   |   |   |
| C’                   | 3 | 1 | 3 | 1 |   | 2 |   |   |   |
| D’                   |   |   |   |   | 1 |   | 3 |   |   |
| E’                   | 2 |   |   |   |   |   |   | 1 |   |
| F’                   | 1 | 2 |   |   |   |   |   |   |   |
| G’                   | 1 | 1 |   |   |   |   |   |   |   |
| H’                   | 5 | 5 | 3 | 2 |   |   |   |   |   |
| I’                   | 8 | 6 | 4 |   |   |   |   |   |   |

was compared to two clustering sets: (1) the reference clustering and (2) the grid-based descriptor in Rodrigues et al. (2017)’s study.

4.3 Clustering evaluation method

In addition to using the reference set for application, we targeted comparing our SC-KM results to the clustering outcome of the grid-based shape descriptor of Rodrigues et al.’s study. The rationale for this comparison is that it is a similar method to our grid-based shape description method. It is important to note that the comparison with Rodrigues et al.’s work was specifically targeted for the fixed aspect ratio case of their grid-based shape descriptor as it is the upmost corresponding method to the grid-based description method we used. The emphasis was solely on grid-based shape descriptors, with no reference for the other available shape descriptors. The study does not suggest a general argument for the present algorithm’s applications in shape description, but it does advance research on grid-based shape descriptors. In our shape comparison algorithm, the aspect ratio has not been changed. Shapes were only uniformly scaled to match the number of cells for our pair-wise comparison, while transformation of aspect ratio (nonuniform scale), rotation, and symmetry were not performed before comparison. This led to variance in the rotated, mirrored, and nonuniformly scaled shapes. In our shape comparison, an L-Shape is different from its rotated L-Shape variant. However, these additional transformation procedures can be incorporated to the method in future developments.

To evaluate the resulted clustering set against the reference set, a clustering accuracy calculation was utilized. The most common method to compute clustering accuracy is to calculate the percentage of the data that has been correctly clustered against reference data (Story & Congalton, 1986). This calculation is often done using an error or a confusion matrix, also called contingency table, that can be represented as a table of the clustered data comprised of columns, as the reference data and rows of the clustered set under analysis (Story & Congalton, 1986; Vinh et al., 2010) or vice versa. Considering the Confusion Matrices in Tables 1-a, 1-b, and 1-c, Table 1-a represents the comparison of the clustering results of the fixed aspect ratio case of the grid-based (GB) descriptor of Rodrigues et al. (2017) to the reference clustering, and Table 1-b is the comparison of the clustering results of Scenario 1 against the same reference clustering, while Table 1-c depicts the comparison of Scenario 2 against the reference set.

Each matrix depicts how the shapes in clusters (A to I) in the clustering subsets have been dispersed in relation to (A’ to I’) in the reference clustering set. The values in each cell were determined as the number of corresponding shapes of the clustering subset to the reference. In the confusion matrix of Table 1-b, for instance, in the column of Cluster A, two shapes correspond with the A’, three shapes belong to Cluster D’, and two belong to Cluster F’. Similarly, all the other columns were organized, via scattering the shapes according to their reference clusters. The cell of highest value in each column has been shaded. For the accuracy calculation, the shaded boxes are considered for the SUM function.

The methods used for evaluating clustering in Rodrigues et al.’s study of accuracy, and the Rand Index were also pursued here. The calculation of the overall level of accuracy in the above confusion matrices is performed by dividing the sum of the highest values in each column by the total number of the reference data. This accuracy measure of each cluster was computed.
using the following formula:

\[
\text{Accuracy} = \frac{\text{SUM}_{\text{highest value of each column}}}{\text{SUM}_{\text{matrix}}}.
\] (1)

As a result, the accuracies in Matrices (1-a), (1-b), and (1-c) were 55.5%, 62.5%, and 65.2% respectively. Further, the calculation of the Rand Index was considered. Developed by Rand in statistics, the index is particular for measuring data clustering by calculating the similarity between two sets of clusterings (Rand, 1971). For Rand (1971), evaluation of a clustering method requires either comparing its results to standard results or to another result. In addition to those two metrics, other clustering evaluation metrics were calculated for additional assessments. One of the metrics that is often used for clustering evaluation is Precision, which can also be called the confidence value that “denotes the proportion of predicted positive cases that are correctly real positives” (Powers, 2011, p. 2). Another important measure for assessing clustering is the Recall or Sensitivity metric, which is used to identify the rate of real positive items that are correctly predicted positive. The measure considers the ratio of the True Positives over the total amount of items that are True Positives and False Negatives (Powers, 2011). Calculating those two measures facilitates the retrieval of one more metric called the F1-Score or F-measure, which is the harmonic average of both measures Precision and Recall (Powers, 2011). When F1-Score reaches its maximum value at 1, this means perfect precision and recall are attained (Sasaki, 2007). These three metrics are calculated as follows (Powers, 2011):

\[
\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}
\] (2)

\[
\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}
\] (3)

\[
F1 = \frac{(2 \times \text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}.
\] (4)

Calculating the Rand Index, Precision, Recall, and F1-Score was performed through implementing the Python program-based algorithm of Tom (2014) that has been developed according to Manning et al. (2008)’s work. Following this method, evaluating Scenario 1, Matrix 1-b gives Rand Index of 0.85, Precision of 0.44, Recall of 0.41, and F1-Score of 0.42 as shown in Table 2-b, compared to 0.82, 0.37, 0.28, and 0.32, respectively, for Matrix 1-a of the Rodrigues et al. (2017)’s grid-based results of Table 2-a. The evaluation metrics of the 64-cell packing resulted in values of Rand Index of 0.85, Precision of 0.48, a Recall measure of 0.38, and F1-Score of 0.42, as depicted in Table 2-c.

It is important to emphasize here that both the shape description method and the clustering method of this work are different from Rodrigues et al.’s work. In their shape description method, Rodrigues et al. have used the grid’s binary vector as a matrix where each matrix contains the corresponding values of the overlaid grid. In our grid-based shape comparison, an exhaustive search for an optimum overlap enabled by the Hungarian algorithm in the pair-wise shape comparison was pursued. The other difference is the clustering method. In their work, the Ward linkage clustering method was used for grouping the shapes, while we used K-Medoids clustering. Those two differences can explain the reason for the different clustering results we retrieved, in comparison with Rodrigues et al.’s results.

### 4.4 Randomness in K-Medoids clustering

It is important to signify that our clustering results discussed above have been retrieved in one instance of running the K-Medoids clustering method, with 1000 iterations. When rerunning the algorithm, different clustering results emerge. For Scenario 1, eight cases of different clustering results were retrieved from running the algorithm eight times, with a range of accuracy between 42/72 and 47/72. Two examples of those emerged clustering results are illustrated in Fig. 7 with accuracy measures of 43/72 = 59.7% for the upper clustering, and 45/72 = 62.5% for the lower clustering, respectively.

In the second scenario of 64-cell packing, the 8 times algorithmic run resulted in a range of accuracy measures between 42/72 and 47/72. Two of these results are depicted in Fig. 8 with
accuracy measures of $45/72 = 62.5\%$ for the upper clustering, and $43/72 = 59.7\%$ for the lower clustering, respectively. The similarities of the cluster subsets between the two results are noticeable, and in some cases, such as clusters D and I in the top image, the subsets are identical to clusters C and E in the bottom image, respectively. This leads to the assertion that despite randomness, the results may converge to the optimum assignment of shapes to their medoids, in grouping the clusters.

5. Discussion of Results
In this test-case application, the SC-KM method was further improved, incorporating a packing algorithm for grid-based description of the selected shapes. In addition, and more importantly, clustering evaluation and external validation were conducted. The accuracy measure and all the four additional clustering measures used show higher values than the compared
Figure 8: Two sample clustering results of the 64-cell-packing scenario that emerge from running the K-Medoids clustering algorithm twice.

study for the two carried out scenarios, as illustrated in the cases shown in the tables. Despite that the reference set does not necessarily represent the best clustering since it was created by human examiners typifying the shapes into subsets of typological characteristics without the use of an algorithmic clustering (Rodrigues et al., 2017), the overall consistency of the clusterings can be considered satisfactory, with the existence of outliers. Qualitatively, perceptual coherence, a visual measure of the clustering consistency where shapes of similar geometric features are clustered together, was also attained.

Grid-based descriptor is a widely used 2D shape descriptor and could be extended to a voxel-based model that is a basic 3D shape representation with benefits such as improved robustness with respect to polygonal surface variations (Zhang et al., 2007). Our algorithm has been tested on a grid-based 2D shape descriptor, with the potential to be applied to a voxel-based 3D
shape descriptor for architectural applications. Similar 2D to 3D grid-based approach has been demonstrated in the research of path planning algorithms (Carsten et al., 2006). In addition, the clustering of shapes for other 2D and 3D shape descriptors will be investigated in future work.

An assertion can be drawn from the results, which is consistent with the grid-based descriptor’s characteristics found in research, that proportions are of high importance to determine shape similarity and difference. Yet, typological shapes can still be successfully traced in the clustering patterns. Overall, as confirmed by research, the K-Medoids clustering methods, in general, perform better in comparison with other methods (Jayanti et al., 2009). In our work, the investigation started with K-Means clustering, but after experimentation, K-Medoids proved to be more successful due to its dependence on the medoid as a central cluster member, which served as a representative clustering shape. We aim to explore other clustering methods in future work. As a limitation, one of the characteristics of the K-Medoids clustering method is randomness of establishing the initial methods, leading to new results re-executing the algorithm, which can be considered problematic. However, for design space exploration, this randomness may not cause an issue, since the algorithm always progresses to convergence, and performs satisfactory clustering.

6. Conclusions and Future Work

The research project marks one of the early attempts to develop a shape clustering method incorporated into a comprehensive GDS, demonstrated by a prototype that is general enough and applicable to a range of design problems. In this paper, a demonstration of evaluating the developed shape clustering method was targeted. Application of the SC-KM method to a new set of shapes has led to its further development and addition of a packing method. For next steps, empirical studies are needed for assessing the effectiveness of user application of the SC-KM method into generative design workflows.

Pursuing evaluation metrics to compare the method to another study provided quantitative analytics and external validation. Those clustering evaluation metrics showed slightly higher values, yet it is expected that further improvement to the shape comparison method can lead to improved results. Overall, pursuing the evaluation of ML-based strategies becomes significant in advancing those ML methods, and necessitates further investigation.

Throughout developing and experimenting with the algorithms and tools of the SC-KM, some limitations have been identified. One of the limitations is the computation time needed, particularly for running the pair-wise shape difference analysis. Approaches to resolve the computing load problem are in progress. Currently, the SC-KM method leads to shape clustering with invariance to scaling and translation, yet with variance to rotation and reflection. Thus, the method can be developed to lead to the invariance of rotation and reflection. The SC-KM method can be used to cluster 2D architectural elevations, sections, and other 2D shapes. Extending the application to 3D forms using voxel description is targeted as a further development to enable wide-ranging applications. Therefore, future work includes applying the method to 3D forms, yet this requires a significant development to the clustering mechanism. Another significant development to this work involves investigating the integration of ML techniques to automatically perform real-time clustering, after training the model with a large dataset of shape-cluster pairs.

The aim of this research was to develop and explore mechanisms for making sense of the generated design set in terms of form/shape evaluation, disrupting existing generative protocols. The main idea is that reviewing qualities of generatively emerging designs is essential for the design process and thus needs to be integrated into the GDS frameworks. The underlying argument in this research is that designers’ agency should be facilitated for enhanced interaction with such generative systems. Further research needs to be done in this area to achieve successful human–machine collaboration.

Acknowledgment

This work has been partially developed within the graduate study supported by multiple scholarships from the Department of Architecture, Texas A&M University, and an internal seed grant by the College of Arts and Letters at Florida Atlantic University.

Conflict of interest statement

None declared.

References

AbdelRahman, M. (2017). GH_CPython: CPython plugin for grasshopper. https://doi.org/10.5281/zenodo.888148.
Abdelrahman, M. M., & Toutou, A. M. Y. (2019). [ANT]: A machine learning approach for building performance simulation: Methods and development. The Academic Research Community Publication, 3(1), 205–213.
Ayorkor, M.-T., Anthony, S., & Bernardine, D. (2007). The dynamic Hungarian algorithm for the assignment problem with changing costs.
Bailey, K. (1994). Numerical taxonomy and cluster analysis. In M. S. Lewis-Beck (Ed.), Typologies and taxonomies: An introduction to classification techniques. https://doi.org/10.4135/9781412986397.
Barnes, M. R. (1999). Form finding and analysis of tension structures by dynamic relaxation. International Journal of Space Structures, 14(2), 89–104. https://doi.org/10.1260/0266351991494722.
Brown, N. C., & Mueller, C. T. (2019). Quantifying diversity in parametric design: A comparison of possible metrics. Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM, 33(1), 40–53. https://doi.org/10.1017/S089906418000033.
Cai, C., & Li, B. (2020). Cluster analysis for urban morphological analysis and case-based design. Paper presented at the ACADIA 2020: Distributed Proximities. Proceedings of the 40th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA).
Carsten, J., Ferguson, D., & Stentz, A. 2006. 3d field d: Improved path planning and replanning in three dimensions. In 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems.
Cha, M. Y., & Gero, J. S. (1998). Shape pattern recognition using a computable pattern representation. In Artificial Intelligence in Design’98.
de las Heras, L.-P., Fernández, D., Fornés, A., Valveny, E., Sánchez, G., & Lladós, J. (2013). Runlength histogram image signature
for perceptual retrieval of architectural floor plans. In International Workshop on Graphics Recognition.

Dutta, A., Lladós, J., Bunke, H., & Pal, U. (2013). A product graph based method for dual subgraph matching applied to symbol spotting. In International Workshop on Graphics Recognition.

Han, J., Pei, J., & Kamber, M. (2011). Data mining: Concepts and techniques. Elsevier.

Harding, J., & Brandt-Olsen, C. (2018) Biomorpher: Interactive evolution for parametric design. International Journal of Architectural Computing, 16(2), 144–163, 10.1177/147807711878579, 1478-0771

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. ACM Computing Surveys (CSUR), 31(3), 264–323.

Jayanti, S., Kalyanaraman, Y., & Ramani, K. (2009). Shape-based clustering for 3D CAD objects: A comparative study of effectiveness. Computer-Aided Design, 41(12), 999–1007.

Korf, R. E. (2002). A new algorithm for optimal bin packing. Paper presented at the 18th National Conference on Artificial Intelligenceda.

Korsah, G. A., Stentz, A., & Dias, M. B. (2007). The dynamic Hungarian algorithm for the assignment problem with changing costs. Retrieved from Carnegie Mellon University.

Li, P. (2020). Deep clustering and morphological analysis of campus context based on a convolutional autoencoder. Paper presented at the ACADIA 2020: Distributed Proximities. Proceedings of the 40th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA).

Manning, C., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge University Press.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. Journal of the Society for Industrial and Applied Mathematics, 5(1), 32–38. Retrieved from http://www.jstor.org/stable/2098689

Nejur, A., & Steinfeld, K. (2016). Ivy: Bringing a weighted-mesh to user’s perspective. Photogrammetric Engineering and Remote Sensing, 52(3), 397–399.

Rand, W. M. (1971). Objective criteria for the evaluation of clusterings. Journal of the American Statistical Association, 66(336), 846–850.

Rodrigues, E., Gaspar, A. R., & Gomes, A. (2013). An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture, Part 1: Methodology. Computer-Aided Design, 45(5), 887–897.

Rodrigues, E., Sousa-Rodrigues, D., de Sampaio, M. T., Gaspar, A. R., Gomes, A., & Antunes, C. H. (2017). Clustering of architectural floor plans: A comparison of shape representations. Automation in Construction, 80, 48–65. https://doi.org/10.1016/j.autcon.2017.03.017.

Sajjanhar, A., & Lu, G. (1997). A grid-based shape indexing and retrieval method. Australian Computer Journal, 29(4), 131–140.

Sasaki, Y. (2007). The truth of the F-measure. In Teaching, tutorial materials (Vol. Version: 26th, pp. 1–5).

Story, M., & Congalton, R. G. (1986). Accuracy assessment: A user’s perspective. Photogrammetric Engineering and Remote Sensing, 52(3), 397–399.

Toffolo, A., & Benini, E. (2003). Genetic diversity as an objective in multi-objective evolutionary algorithms. Evolutionary Computation, 11(2), 151–167.

Tom. (2014). Rand index calculation. https://stats.stackexchange.com/q/110712. Retrieved fromhttps://stats.stackexchange.com/users/23823/tom, Access date: April 28th 2021.

Turrin, M., Yang, D., D’Aquilio, A., Sileryte, R., & Sun, Y. (2016). Computational design for sport buildings. Procedia Engineering, 147, 878–883.

Velmurugan, T., & Santhanam, T. (2010). Computational complexity between K-means and K-medoids clustering algorithms for normal and uniform distributions of data points. Journal of Computer Science, 6(3), 363–368.

Vinh, N. X., Epps, J., & Bailey, J. (2010). Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. The Journal of Machine Learning Research, 11, 2837–2854.

Wagstaff, K., Cardie, C., Rogers, S., & Schrödl, S. (2001). Constrained k-means clustering with background knowledge. Paper presented at the ICML ’01 Proceedings of the 18th International Conference on Machine Learning.

Ward, J. H. Jr, (1963). Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association, 58(301), 236–244.

Wilks, D. S. (2011). Cluster analysis. In International geophysics(Vol. 100, pp. 603–616). Elsevier.

Yousif, S., & Yan, W. (2019a). Application of an automatic shape clustering method into generative and design optimization systems. Paper presented at the ACADIA 19: Ubiquity and Autonomy [Proceedings of the 39th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA)] ISBN 978-0-57-59179-7, The University of Texas at Austin School of Architectures.

Yousif, S., & Yan, W. (2019b). Shape clustering using K-Medoids in architectural form finding.

Yousif, S., Yan, W., & Culp, C. (2017). Incorporating form diversity into architectural design optimization. Paper presented at the ACADIA 2017: Disciplines & Disruption. Proceedings of the 37th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA), MIT. http://papers.cunicad.org/cgi-bin/works/paper/acadia17_640.

Zhang, D., & Lu, G. (2004). Review of shape representation and description techniques. Pattern Recognition, 37(1), 1–19.

Zhang, L., da Fonseca, M. J., & Ferreira, A., & Combinando Realidade Aumentada e Recuperação. (2007). Survey on 3D shape descriptors. Fundação para a Ciência e Tecnologia, Lisboa, Portugal, Tech. Rep. Technical Report, DecorAR (FCT POS/EIA/59938/2004)(p. 3).

Zwierzyci, M., Nicholas, P., & Thomsen, M. R. (2018). Localised and learnt applications of machine learning for robotic incremental sheet forming. In K. De Rycke, C. Gengnagel, O. Baverel, J. Burry, C. Mueller, M. M. Nguyen, P. Rahm, & M. R. Thomsen (Eds.), Humanizing digital reality: Design modelling symposium Paris 2017(pp. 373–382). Springer.