# Enhanced Convex Hull based Clustering for High Population Density Avoidance under D2D Enabled Network

Vishaka Basnayake[1,2], Hakim Mabed[1], Philippe Canalda[1], Dushantha Nalin K. Jayakody[2,3]
[1]*DISC/FEMTO-ST, Univ. Bourgogne Franche-Comte,* Cours Louis Leprince-Ringuet, 25200, Montbeliard, France
[2]*Centre for Telecommunication Research, School of Engineering, Sri Lanka Technological Campus,* 10500 Padukka, Sri Lanka
[3]*School of Computer Science and Robotics, Tomsk Polytechnic University,* 634050 Tomsk, Russia
{vishaka.basnayake_mudiyanselage, hakim.mabed, philippe.canalda}@univ-fcomte.fr, dushanthaj@sltc.ac.lk

***Abstract*—Global pandemics such as Covid-19 have led to massive loss of human lives and strict lockdown measures worldwide. To return to a certain level of normalcy, community awareness on avoiding high population density areas is significantly important for infection prevention and control. With the availability of new telecommunication technologies, it is possible to provide highly informative population clustering data back to people using wireless aerial agents (WAAs) placed in a local area. Hence, a service architecture that allows users to access the localization of population clusters is proposed. Further, a convex hull-based clustering method, enhanced population clustering (*E-PC*), is proposed. This method refined the result of conventional clustering methods such as *K-means* and *Gaussian mixture model (GMM)*. Moreover, the potential in *E-PC* to achieve the same or higher results compared to the original *K-means* and *GMM*, while consuming lesser data points, is demonstrated. On average, *E-PC* improved the cluster detection performance in both *K-means* and *GMM* by 18.93% under different environments such as remote, rural, suburban, and urban in terms of silhouette score. Further, *E-PC* allows a 15% data reduction which results in decreasing the computational cost and energy consumption of the WAAs.**

***Index Terms*—pandemic prevention, population clustering, convex hull, K-means, Gaussian mixture model**

## I. Introduction

Crowd monitoring is an active research area that is used for different applications such as behavior analysis, intelligent surveillance systems, public riots detection, and congestion avoidance [1]. A high population congestion is one of the main reasons for infectious diseases such as Covid-19 to spread fast and over a large geographical area [2]. Physical distancing has been argued as one of the effective means to combat the spread of such pandemics before vaccines or therapeutic drugs becomes available. Thus, the knowledge on population cluster centers in local areas, e.g., small cities, shopping complexes, schools, is important in avoiding over-crowded areas and preparing for current and future pandemics [3]. A variety of wireless communication and positioning technologies, including drones, cellular positioning systems, and global positioning systems (GPS), can be used to monitor the crowd gatherings outdoors [4].

Table I: The list of symbols and notations used in this paper

| Symbol | Description |
|---|---|
| $N_l$ | Number of total UE locations,$N_l$ |
| $\chi$ | Convex hull of the total UE locations |
| $\varepsilon^*$ | Maximum volume inscribed ellipse inside $\chi$ |
| $\mathcal{B}$ | Deformation matrix from unit circle to an ellipse |
| $\boldsymbol{d}$ | Vector representing the orientation of the ellipse |
| $\rho_2$ | Volume of a planar unit circle |
| $K_{opt}$ | Optimum number of clusters |
| $M$ | Set of 4G/5G D2D enabled UEs |
| $c_c$ | Cluster centers |
| $u_a$ | Wireless aerial agent (WAA) that computes $c_c$ locations |
| $T_w$ | Periodic time delay of updating location data to the $u_a$ |
| $T_0$ | Maximum time period of storing the received location data |
| $T_M$ | Fixed time period used to compute the average number of requests |
| $RR$ | Average number of requests per $T_w$ during the last $T_M$ period |
| $d_{in}$ | Total number of data points in the dataset of UE locations |
| $d_{\varepsilon^*}$ | Number of data points which are only inside the $\varepsilon^*$ |
| $d_r$ | Data size reduction |
| $d_{r_m}$ | Maximum data size reduction possible |

Further, D2D introduced in 4G enables UEs to establish a direct communication link with another UE [5]. Thus, D2D technology allows wireless aerial agents (WAAs) to reach UEs in the network edge or out of coverage areas, which may not be captured under the cellular network infrastructure [6].

Techniques used for crowd monitoring could be mainly classified into supervised or unsupervised approaches. Supervised learning methods are used for applications such as behavior detection, density estimation, etc. Unsupervised learning methods are used for crowd clustering where clusters are detected via distance-based algorithms [7]. Meanwhile, recent trends in convex hull based methods for clustering have paved the way to simplify and enhance the performance in

clustering algorithms [8].

**The main contributions of this work are**:

- Propose a service architecture that allows users to access the localization of population clusters
- Provide a novel method to improve population cluster detection and decrease computational cost in a local D2D network by adopting the maximum volume inscribed ellipse in a convex hull (MVIE) principle [9] in unsupervised learning.
- Comparison of the proposed method with two baseline clustering methods: *K-means* and *GMM*, in network regions with different population sizes such as remote, rural, suburban, and urban.

The remainder of this paper is organized as follows, Section II provides an overview of other related works, while Section III details the system model and the design of the proposed algorithm. In Section IV, the performance analysis of the method is conducted. Further, Table I provides the list of symbols and notations applied in this paper.

## II. Related work

Clustering is a data processing algorithm that classifies a set of entities based on proximity between cluster members, the concentration zones of the data, and their statistical distributions [10]. The purpose of clustering is both to minimize the intra-cluster distance and to maximize the inter-cluster distance.

*K-means* and *GMM* are two of the most conventional clustering methods [11]. *K-means* approach groups data into clusters by minimizing the distances between data points and their cluster's centroid [12]. To that end, in iterative way, $k$ initial points ($k$ is an input parameter of *K-means* method) from the dataset are randomly selected as cluster centers. Then the other dataset points are assigned to the closest cluster center. The sum of the intra-cluster distance is then computed. The process is repeated a given number of iterations to minimize the sum of intra-cluster variations.

In contrast, *GMM* identifies clusters using an expectation-maximization (EM) algorithm assuming that all data points are generated from a mixture of a finite number of Gaussian distributions. Further, it has been shown that *GMM* is a complex classification method and has a higher computation time compared to *K-means* [13]. Hence, *GMM* may be less energy efficient, limiting its use for low computing capacity in WAAs. The unawareness of the number of clusters in the input data represents one of the issues with the unsupervised learning clustering approaches. Several techniques such as elbow method, silhouette method and gap statistics method have been proposed to compute the optimal number of clusters in the data [14].

Comparative studies show that *gap statistics method* provides better performance concerning the identification of the optimal number of clusters [15]. In *gap statistics method* different values for the number of clusters, $k$ are evaluated. For each $k$ value, the clustering algorithm is run, then the sum of intra-cluster variation, $W_k$, over the generated clusters is computed. Thereafter, a set $B$ of uniformly distributed points are generated and clustered. Finally, sum of intra-cluster variation over clusters of the set $B$, $W_k^*$, is computed and compared with $W_k$ [16]. Hence, gap statistics for a particular $k$, is mathematically defined as:

$$Gap_k = log(W_k^*) - log(W_k), \tag{1}$$

Finally, the $k$ value that maximizes $Gap_k$ is selected as the optimal $k$.

Meanwhile, research works in [8], [17] have proposed methods to incorporate convex hull principles to improve performance in clustering algorithms. Convex hull of a planar set of points $P$ is the smallest convex set enclosing $P$ [18]. The convex hull is obtained in practice using facet enumeration techniques. Quickhull algorithm is such a facet enumeration method, which is considered an efficient and practical method for convex hull computation [19].

The Quickhull algorithm is based on the divide-and-conquer algorithm that divides the input data into subsets and recursively evaluate such subsets [20]. It is started by first computing the maximum and minimum points of $P$. Secondly, the horizontal and vertical lines passing through these points are found to form a bounding rectangle, which forms the initial hull. Next, the points lying within such a quadrilateral are eliminated from further consideration. Finally, as the algorithm executes, a convex hull is formed once the set of points outside each edge of the polygon is zero [19].

In addition, by adopting the maximum volume inscribed ellipse in a convex hull (MVIE) principle [9] in unsupervised learning, an understudied method of population clustering in a local D2D network is proposed.

## III. Methodology

In this paper, a novel method for population cluster detection called enhanced population clustering (*E-PC*) under a local D2D network, where users' phones reside within the coverage range of the WAA, is proposed. A MVIE based *K-means* clustering is used in *E-PC* to identify more than one cluster.

Let $M$ be a set of 4G/5G D2D enabled UEs that have subscribed to the population clustering service provided by a WAA, namely $u_a$, which is hovering above a specific height in the local area (Figure. 1). Each $m \in M$ utilizes a physical sidelink control channel (PSCCH) and a physical sidelink shared channel (PSSCH) to establish a D2D link with $u_a$. Two D2D modes are distinguished according to the coverage status of the user device and WAA [21]. In in-coverage D2D communication mode, one of the two devices is under the coverage of a given gNB station. However, in out-of-coverage mode [22], the synchronization of the two devices is made using the primary synchronization signals (PSSs) and secondary synchronization signals (SSSs).

Once a D2D link is established with the $u_a$, the device $m \in M$ transmits its location information to $u_a$ every $T_w$ period. The $u_a$ stores the received location data for a maximum time period $T_0 > T_w$. If the location data of a given mobile

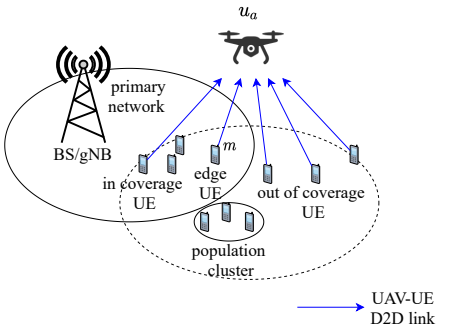


Figure 1: WAA $u_a$ collecting location data from $N_l$ number of UEs in a local region to identify the population clusters using the proposed algorithm.

$m$ is not updated after the $T_0$ period (e.g. the mobile switches off), the location data of the mobile is removed.

The $u_a$ works in two modes according to the clustering requests rate. The $u_a$ counts the number of requests received each $T_w$. In each period $T_M$, the $u_a$ computes the average number of requests per $T_w$ during the last $T_M$ period, $RR$. If this average exceeds 1, then the $u_a$ switches to the *periodic clustering mode*. While if the requests rate $RR$ is lower than 1, the $u_a$ uses the *on-demand clustering mode*.

In the *periodic clustering mode*, the $u_a$ computes the cluster centers in each $T_w$. Once a clusters' localization request is received, $u_a$ sends the coordinates of the last computed cluster centers. Since the number $RR$ is higher than 1, the *periodic mode* allows to limit the amount of energy consumed by $u_a$. In contrast, in *on-demand mode*, the clusters are only computed when a new request is received. Consequently, the $u_a$ does not waste energy by computing the clusters when there is no request.

To compute the cluster centers $c_c$, $u_a$ uses the last received location data, $N_l$, from $M$ during the last $T_0$ period. The mobile device localization is referenced by its geographical coordinates: longitude and latitude. Once a clusters' centers request is received, the response is sent to the user device $m$ through the D2D sidelink channels. The clustering procedure in both *on-demand mode* and *periodic mode*, involves three main steps: convex hull computation, maximum volume inscribed ellipse optimization, and cluster centers computation.

### A. Convex hull computation

The convex hull of the set $N_l$ is computed by calling a facet enumeration algorithm called Quickhull algorithm [19]. The convex hull of a set of points corresponds to the smallest convex 2D area that covers the set of points (see Figure 3). Formally, the convex hull area of $N_l$, $\chi$, contains the set of points, $\mathbf{x}$, respecting the following $p$ inequalities:

$$\chi = \{\mathbf{x} \in \mathbb{R}^2 \mid \boldsymbol{a_i^T}\mathbf{x} \leq \boldsymbol{b}_i,\ i = 1, ..., p\}, \tag{2}$$

Each equation $a_{i,1} \times x + a_{i,2} \times y = b_i$ corresponds to a line segment forming the convex hull boundary of $N_l$, where $a \in \mathbb{R}^{2\times p}$, $b \in \mathbb{R}^p$.

### B. Maximum volume inscribed ellipse optimization

During the second step of the clustering procedure, $u_a$ computes the ellipse included in the convex hull with the maximum volume. Let's assume that the maximum volume inscribed ellipse, inside $\chi$ (Figure 3) contains the major part of the localized mobiles $M$. Therefore, regions outside the ellipse while being inside the convex hull represent negligible clustering areas. The area inside an ellipse $\varepsilon$ could be defined as follows:

$$\varepsilon = \{\mathbf{x} \mid \mathbf{x} = \mathcal{B}\boldsymbol{u} + \boldsymbol{d}, \boldsymbol{u} \in \mathbb{R}^2, \|\boldsymbol{u}\|_2 \leq 1\}, \tag{3}$$

$\mathcal{B} \in \mathbb{R}^{2\times 2}$ is a symmetric positive definite matrix that represents the deformation matrix from a unit circle to an ellipse and $\boldsymbol{d} \in \mathbb{R}^2$ is the vector that gives the orientation of the ellipse.

Maximum volume inscribed ellipse $\varepsilon^*$ is obtained by determining the ellipse parameters, $B$ and $d$, that maximizes the ellipse volume given in (5), and ensures that the ellipse is included in the convex hull ($\varepsilon^* \subseteq \chi$). Such constraint can be represented as the set of inequalities provided in (4).

$$\|\mathcal{B}\boldsymbol{a}_i\| + \boldsymbol{a}_i^T\boldsymbol{d} \leq \boldsymbol{b}_i,\ i = 0, ..., p. \tag{4}$$

The volume of an ellipse with parameters $B$ and $d$ is given by:

$$Vol(\mathcal{B}, \boldsymbol{d}) = \rho_2\sqrt{det(\mathcal{B}^T\mathcal{B})}, \tag{5}$$

$\rho_2$ gives the volume of a planar unit circle [9]. Hence, the maximum volume inscribed ellipse optimization is formulated as follows:

$$\begin{aligned} \min_{\mathcal{B},\boldsymbol{d}} \quad & -\sqrt{det(\mathcal{B}^T\mathcal{B})} \\ \text{s.t.} \quad & \|\mathcal{B}\boldsymbol{a}_i\| + \boldsymbol{a}_i^T\boldsymbol{d} \leq \boldsymbol{b}_i,\ i = 0, ..., p \\ & \mathcal{B} \succeq 0. \end{aligned} \tag{6}$$

Noting that the objective function $det(\mathcal{B}^T\mathcal{B})$ in (6) is a non-convex function. The non convexity of the objective function leads to some problem of optimization convergence. To overcome this problem, the maximum volume inscribed ellipse is reformulated as a negative log determinant maximization problem [9], [18]. Log of determinant is used to ensure convexity of the objective function. This new problem is formulated as follows:

$$\begin{aligned} \min_{\mathcal{B},\boldsymbol{d}} \quad & -\log\det(\mathcal{B}) \\ \text{s.t.} \quad & \|\mathcal{B}\boldsymbol{a}_i\| + \boldsymbol{a}_i^T\boldsymbol{d} \leq \boldsymbol{b}_i,\ i = 0, ..., p \\ & \mathcal{B} \succeq 0. \end{aligned} \tag{7}$$

In this work, CVXOPT solver was used to solve the MVIE optimization problem given in (7). CVXOPT is a convex optimization solver based on the Python programming language.

### C. Cluster centers computation

The goal is to detect cluster centers formed by the UEs and consequently avoid dense zones. It was assumed that UEs form clusters around the cluster centers following a Gaussian multivariate density function [23] with a $\mu$ 2D mean vector, and $\mathbf{\Sigma}^2$ covariance matrix, given by (8). The graphical representation of UE locations following such a Gaussian density function is illustrated in Figure. 2.

$$f(x|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)|\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2}(x-\mu)^T\mathbf{\Sigma}^{-1}(x-\mu)\right), \tag{8}$$

where $x$ denote the coordinates of the UE locations.

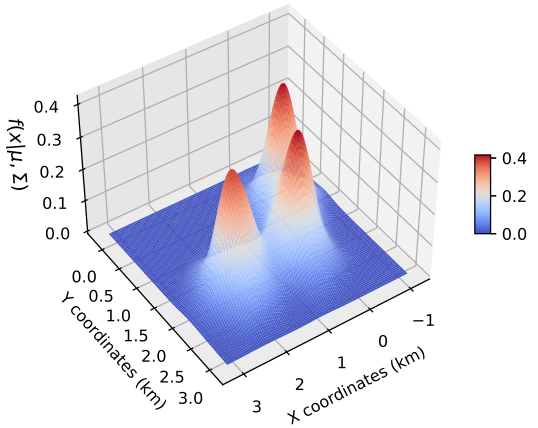


Figure 2: UEs forming clusters around three centers following a Gaussian multivariate density function, given in (8), each with a cluster variance of 0.3.

The UEs with a location outside the maximum volume inscribed ellipse $\varepsilon^*$, are more sparse and may decrease the clustering performance. Hence, in our proposed method, *K-means* is applied only on the locations inside $\varepsilon^*$.

The optimal number of clusters $K_{opt}$ inside $\varepsilon^*$ are computed using the gap statistics method [16] discussed in the *related works* section. $K_{opt}$ corresponds then to the cluster number which has the highest gap statistic value. Finally, the set of the $K_{opt}$ cluster centers $c_c$ are detected as shown in Figure 3. The location of $c_c$ of each cluster is computed by the arithmetic mean of all data points that belong to that cluster [24].

Once a request is received, $u_a$ sends the set of coordinates of the $c_c$ to the corresponding mobile. The procedure followed by $u_a$ for computing the cluster centers is summarized in Algorithms 1, 2 and 3.

## IV. Performance Analysis

### A. Numerical simulations

The proposed algorithm was implemented in the Google Colab online platform [25]. The simulation parameters used are presented in Table II. For each scenario, a sample dataset was generated containing $n_{a_c}$ isotropic Gaussian clusters with the same variance $S^2$ using the *make blobs* Python function [26].

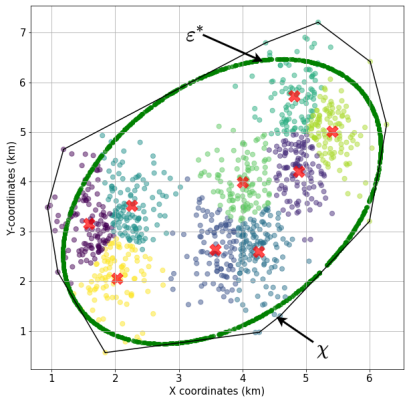


Figure 3: Proposed method: First, the optimal number of clusters, $K_{opt}$, in the UE locations inside the maximum volume inscribed ellipse $\varepsilon^*$ is computed. Next, a conventional clustering method is used to detect $K_{opt}$ in the total UE locations in the network area. Distinct colors represent the different clusters. The red crosses denote the cluster center locations, $c_c$.

**Algorithm 1:** Procedure run every $T_w$ on $u_a$

```
Input UE locations N_l
Output cluster centers c_c
if RR >= 1 then
    for every T_w period do
        Compute convex hull boundary of N_l, χ, locations
        Selecting the locations inside ε* from N_l
        Determine K_opt using gap statistics method
        Use a conventional clustering method to find c_c
    end
end
```

The clustering procedure on the generated dataset was started by calculating the convex hull, $\chi$, of the points. Then using the optimization tool, CVXPY [27], the maximum volume inscribed ellipse, $\varepsilon^*$, was computed. Next, the gap statistics method was applied to determine the optimal number of clusters $K^*_{opt}$. Indeed, since that gap statistics method is not deterministic (see Section II), the obtained $K_{opt}$ can change on different runs. Hence, *gap statistics method* was applied for a given number of iterations, $r_{km}$, and the obtained $K_{opt}$ with the highest gap statistics was selected as the most suitable $K^*_{opt}$. It should be noted, that the gap statistics method was applied only on points of the dataset that are inside the $\varepsilon^*$s. Once $K^*_{opt}$ was computed, a conventional clustering method, i.e., *K-means*, *GMM*, was applied on the $N_l$ points to determine the cluster centers $c_c$.

To ensure the relevance of the comparison, every method was run $r_d$ times using the same simulation parameters but with different datasets. The results shown below correspond to the average evaluation of the method for each scenario.

**Algorithm 2:** Procedure run every $T_M$ on $u_a$

**Input** list of received requests
**Output** average number of requests per $T_w$ during the last $T_M$ period, RR
Compute the average number of requests per $T_w$

**Algorithm 3:** Procedure run on reception of new request

**Input** mobile sender $m$, UE locations $N_l$
**Output** cluster centers $c_c$
**if** $RR < 1$ **then**
    Compute convex hull boundary of $N_l$, $\chi$, locations
    Selecting the locations inside $\varepsilon^*$ from $N_l$
    Determine $K_{opt}$ using gap statistics method
    Use a conventional clustering method to find $c_c$
**else**
    Use the last computed $c_c$ in Algorithm 1
**end**
Communication of $c_c$ to $m$

### B. Evaluation parameters

Silhouette score and data size reduction were used to evaluate the clustering in proposed and baseline approaches. The cluster detection performance was evaluated using the silhouette score [28]. Such a score was computed to study the separation distance between the resulting clusters [29]. Silhouette value of one data point, $p$, is defined as:

$$S(p) = \frac{(b(p) - a(p))}{\max(a(p), b(p))}, \tag{9}$$

where $a(p)$ is the mean distance between $p$ and all other data points in the same cluster called mean intra-cluster distance, and $b(p)$ is the mean distance between $p$ and all the points of the nearest cluster, called mean nearest-cluster distance [30]. Silhouette score computation returns a value between -1 and 1 that measures the quality of the clusters. More specifically, the higher the silhouette score, the better the cluster detection [28].

The data size reduction, $d_r$, was computed as:

$$d_r = d_{in} - d_{\varepsilon^*}, \tag{10}$$

Table II: Simulation parameters

| Parameter | Value |
|---|---|
| Number of clusters in original data, $n_{a_c}$ | 3 |
| Variance of each cluster | 0.12 |
| Network area | $7 \times 7$ km$^2$ |
| Minimum and maximum limits of $K_{opt}$ | [2,...,15] |
| Total UE number (cluster points), $N_l$ | variable [100,1000,$10^4$, $10^5$] |
| Number of iterations, $r_{km}$ | 10 |
| Number of different datasets, $r_d$ | 3 |

where $d_{in}, d_{\varepsilon^*}$ denotes respectively, the total number of data points in the dataset of UE locations and the number of data points that are inside the $\varepsilon^*$.

### C. Results

The performance of the proposed approach *E-PC* applied on conventional clustering methods, was compared with two conventional methods: *K-means* and *GMM* clustering. The difference in the approaches is that, in the conventional clustering, the gap statistics method is applied on the total UE locations dataset, whereas in *E-PC*, the gap statistics method is applied only on points of the dataset that are inside the $\varepsilon^*$.

Figures 4 and 5 present the silhouette score against the variance, $S^2$, of the generated clusters. It can be seen that, when the cluster variance $S^2 = 0.12$, silhouette score improvement is not significant with *E-PC* compared to only *GMM*. In this instance, the percentage of data reduction is 21.5% of the initial $10^4$ data points. Since the sparsity of the data points is low, the impact of the data points situated outside the ellipse is considerable. Hence, when the data points are less sparsely scattered and the data size reduction exceeds more than 15%, the optimal number of data points required for the proposed approach has not been met as described in detail in Section IV-D.

In general, *E-PC* has achieved the same or higher performance compared to the conventional clustering methods while considering only the data inside $\varepsilon^*$ to compute $K_{opt}$. Such performance can also be seen in Figures 6 and 7, where the silhouette score variation against the number of generated clusters is presented. As the number of clusters increase, the existence of clusters inside $\varepsilon^*$ has increased and enabled *E-PC* to detect clusters notably.

The average silhouette scores achieved for different population sizes with original *K-means*, *GMM*, and after applying *E-PC* on such approaches are presented respectively in Figures. 8 and 9. On average, *E-PC* has provided a silhouette score performance improvement by 18.93% compared to using only *K-means* and *GMM*. The ability of cluster detection has increased when the UE locations outside the $\varepsilon^*$ are excluded from consideration. The presence of clusters has become higher inside the $\varepsilon^*$. The selection of UE locations inside the $\varepsilon^*$ in *E-PC* has enabled achieving the same or improved cluster detection using a reduced data size.

### D. Data size and silhouette score trade off

The maximum data size reduction possible ($d_{r_m}$), which withholds the performance of *E-PC*, under different environment types are given in Table. III. On average, concerning all the environment types, the highest possible data reduction while maintaining the proposed approach performance above the baseline methods was 15% out of the total data points. It should be noted that energy consumption is directly proportional to the data size [31]. Hence, the proposed method improves the energy efficiency at $u_a$ significantly, since the data volume used by the proposed approach was lesser than the baseline approaches.

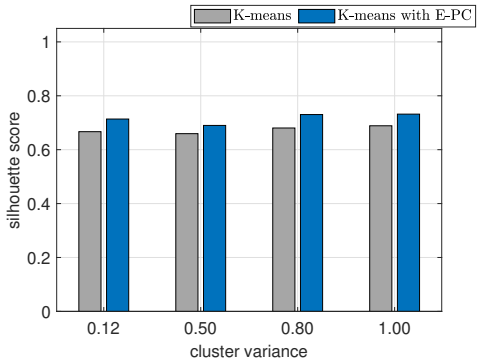


Figure 4: Comparison of silhouette scores obtained with original K-means and after applying E-PC on K-means under several cluster variances in a sub-urban region with 10,000 UEs.

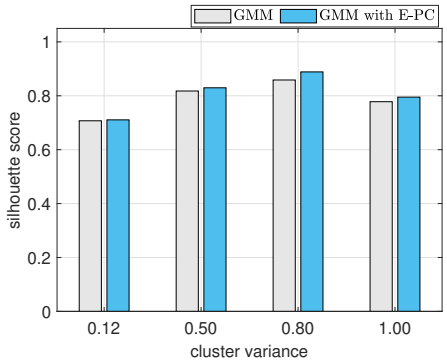


Figure 5: Comparison of silhouette scores obtained with original GMM and after applying E-PC on GMM under several cluster variances in a sub-urban region with 10,000 UEs.

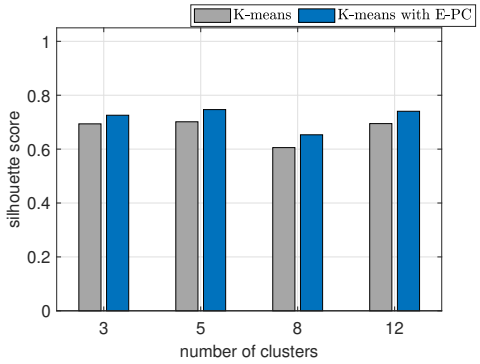


Figure 6: Comparison of silhouette scores obtained with original K-means and after applying E-PC on K-means against number of clusters in a sub-urban region with 10,000 UEs.

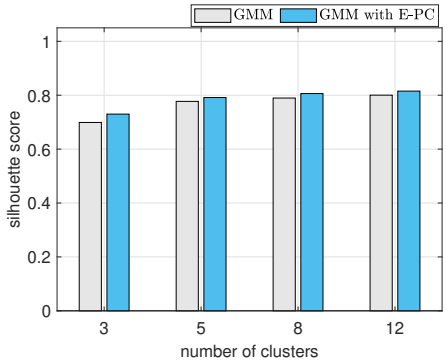


Figure 7: Comparison of silhouette scores obtained with original GMM and after applying E-PC on GMM against number of clusters in a sub-urban region with 10,000 UEs.

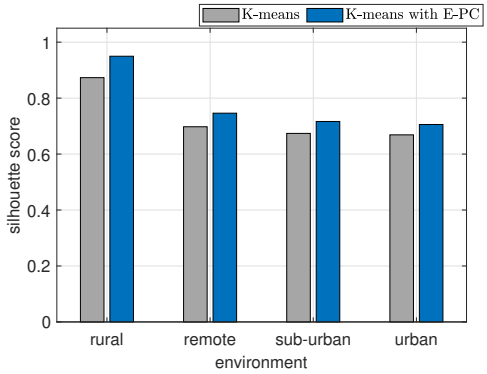


Figure 8: Comparison of silhouette scores obtained with original K-means and after applying E-PC on K-means against several environment types, i.e. rural, remote, sub-urban, urban. It is considered that respectively, rural, remote, sub-urban and urban regions contain $10^2, 10^3, 10^4, 10^5$ UEs.

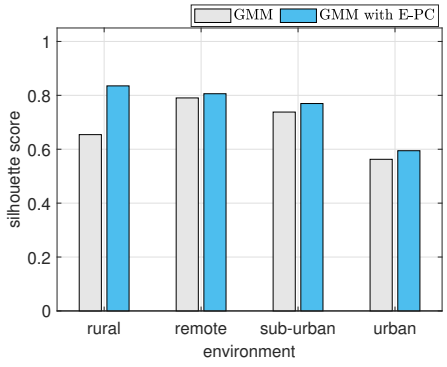


Figure 9: Comparison of silhouette scores obtained with original GMM and after applying E-PC on GMM against several environment types, i.e. rural, remote, sub-urban, urban. It is considered that respectively, rural, remote, sub-urban and urban regions contain $10^2, 10^3, 10^4, 10^5$ UEs.

## V. CONCLUSION

With global pandemics such as Covid-19, awareness of population clusters has increased. First, a service architecture that allows users to access the localization of clusters was pro-

Table III: Data size reduction

| **Environment** | $d_{r_m}$ |
|---|---|
| remote (100 UEs) | 30.33% |
| rural (1000 UEs) | 24.70% |
| sub-urban ($10^4$ UEs) | 25.37 |
| urban ($10^5$ UEs) | 15.24% |

posed. The proposed service works under two different modes: periodic and on-demand mode according to the service load. The objective was to preserve the energy of the wireless aerial agent (WAA) and to increase the system responsiveness. Also a convex hull-based clustering method in local D2D networks was proposed, for population cluster detection. The proposed algorithm considered the region inside the $\varepsilon^*$, to obtain a same or higher performance in cluster detection compared to conventional methods such as *K-means* and *GMM* methods while considering lesser data points than such conventional methods. This clustering algorithm was performed at the WAA that covered the studied local region. The results showed that the proposed method outperformed the original *K-means* and *GMM* clustering performance by 18.93% under different environments such as remote, rural, suburban, and urban in terms of silhouette score. Moreover, the proposed approach allowed a 15% data reduction, thereby reducing the computational cost and energy consumption of the WAA. On the other hand, in the proposed method, UE locations within a cluster followed a Gaussian distribution. Hence, the performance of the proposed method for other distributions of UE locations corresponding to non-open areas, i.e., streets, indoor areas, remains to be evaluated. In addition, our approach is less efficient when the number of points outside $\varepsilon^*$ exceeds a threshold of 15% of the total data. Therefore, the approach can be enhanced by switching to the conventional clustering methods in such a situation.

## REFERENCES


[1] N. Hossein Motlagh, M. Bagaa, and T. Taleb, "Uav-based iot platform: A crowd surveillance use case," *IEEE Communications Magazine*, vol. 55, pp. 128–134, 02 2017.

[2] WHO, "What are the health risks related to overcrowding?" https://www.who.int/water_sanitation_health/emergencies, 2021, (Accessed on 05/04/2021).

[3] N. Kadi and M. Khelfaoui, "Population density, a factor in the spread of covid-19 in algeria: statistic study," *Bulletin of the National Research Centre*, vol. 44, 12 2020.

[4] N. Saeed, A. Bader, T. Y. Al-Naffouri, and M.-S. Alouini, "When wireless communication responds to covid-19: Combating the pandemic and saving the economy," *Frontiers in Communications and Networks*, vol. 1, p. 3, 2020. [Online]. Available: https://www.frontiersin.org/article/10.3389/frcmn.2020.566853

[5] C. Y. Wei, *D2D Physical-Layer Design*, 05 2020, pp. 1–25.

[6] P. Kumar, S. Darshi, and S. Shailendra, "Drone assisted device to device cooperative communication for critical environments," *IET Communications*, vol. n/a, no. n/a. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/cmu2.12134

[7] K. Khan, W. Albattah, R. Khan, A. Qamar, and D. Nayab, "Advances and trends in real time visual crowd analysis," *Sensors (Basel, Switzerland)*, vol. 20, 09 2020.

[8] H. Cevikalp, "High-dimensional data clustering by using local affine/convex hulls," *Pattern Recognition Letters*, vol. 128, pp. 427–432, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865519302806

[9] C. H. Lin, R. Wu, W. K. Ma, C. Y. Chi, and Y. Wang, "Maximum volume inscribed ellipsoid: A new simplex-structured matrix factorization framework via facet enumeration and convex optimization," *SIAM Journal on Imaging Sciences*, vol. 11, 08 2017.

[10] "Cluster analysis," https://medium.com/mlearning-ai/cluster-analysis-6757d6c6acc9, January 2021, (Accessed on 04/15/2021).

[11] C. Guyeux, S. Chrétien, G. Bou Tayeh, J. Demerjian, and J. Bahi, "Introducing and comparing recent clustering methods for massive data management in the internet of things," *Journal of Sensor and Actuator Networks*, vol. 8, p. 56, 12 2019.

[12] "What is clustering? clustering in machine learning," https://developers.google.com/machine-learning/clustering/overview, 2020, (Accessed on 04/04/2021).

[13] E. Patel and D. S. Kushwaha, "Clustering cloud workloads: K-means vs gaussian mixture model," *Procedia Computer Science*, vol. 171, pp. 158–167, 2020, third International Conference on Computing and Network Communications (CoCoNet'19). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920309820

[14] S. Nanjundan, S. Sankaran, C. R. Arjun, and G. P. Anand, "Identifying the number of clusters for k-means: A hypersphere density based approach," *CoRR*, vol. abs/1912.00643, 2019. [Online]. Available: http://arxiv.org/abs/1912.00643

[15] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society Series B*, vol. 63, pp. 411–423, 02 2001.

[16] B. Boehmke, "K-means cluster analysis," https://uc-r.github.io/kmeans_clustering, (Accessed on 04/08/2021).

[17] L. Liparulo, A. Proietti, and M. Panella, "Fuzzy clustering using the convex hull as geometrical model," *Advances in Fuzzy Systems*, vol. 2015, 04 2015.

[18] R. L. Graham and F. Frances Yao, "Finding the convex hull of a simple polygon," *Journal of Algorithms*, vol. 4, no. 4, pp. 324–331, 1983. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0196677483900135

[19] C. Barber, D. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, 02 1998.

[20] Z. Jiayin, G. Mei, N. Xu, and K. Yang, "A novel implementation of quickhull algorithm on the gpu," *arxiv*, 01 2015.

[21] U. Kar and D. Sanyal, "A critical review of 3gpp standardization of device-to-device communication in cellular networks," *SN Computer Science*, vol. 1, 10 2019.

[22] R. Rouil, F. Cintrón, A. Ben Mosbah, and S. Gamboa, "Implementation and validation of an lte d2d model for ns-3," 06 2017, pp. 55–62.

[23] P. Ahrendt, *The Multivariate Gaussian Probability Distribution*, 2005.

[24] J. J. Nick Paine, "K-means clustering and gaussian mixture models," https://towardsdatascience.com/clustering-out-of-the-black-box-5e8285220717, February 2021, (Accessed on 05/13/2021).

[25] "Colaboratory," https://colab.research.google.com/notebooks/intro.ipynb, (Accessed on 04/01/2021).

[26] "sklearn.datasets.make_blobs — scikit-learn 0.24.1 documentation," https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html, (Accessed on 04/01/2021).

[27] "Cvxpy 1.1.11 documentation," https://www.cvxpy.org, 2020, (Accessed on 04/01/2021).

[28] "scikit-learn 0.24.2 documentation," https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html, 2007, (Accessed on 05/11/2021).

[29] M. Chaudhary, "Silhouette analysis in k-means clustering," https://medium.com/@cmukesh8688/silhouette-analysis-in-k-means-clustering-cefa9a7ad111, (Accessed on 05/11/2021).

[30] A. Kumar, "Kmeans silhouette score explained with python example," https://dzone.com/articles/kmeans-silhouette-score-explained-with-python-exam, September 2020, (Accessed on 05/11/2021).

[31] S. Kumar, "Comparative analysis in between the k-means algorithm, k-means using with gaussian mixture model and fuzzy c means algorithm," 02 2017.