Comparative Study of Grid-Inverted List Hybrid Indexing Techniques for Moving Objects and Queries

Sulbha Powar(✉) and Ganesh Magar

P. G. Department of Computer Science, SNDT Women's University, Mumbai, India
sulbha_powar@hotmail.com

Abstract. Advancement in GPS technologies and availability of a variety of devices to capture location and other attributes of the objects has led to an enormous development in geo-textual data. Searching through these objects for relevant objects as per the requirement needs efficient indexing technique and searching algorithm. Past queries, Present queries and Future queries are the three types of geo-textual queries. Past queries are responded based on the historical locations of the moving objects stored in the database. Future queries can be answered if the velocity vector of the object is known in advance. But in many real-time applications the position of the objects in future cannot be predicted. In such a scenario capturing movement of the objects and queries in real time and answering queries or updating query answer sets in real time is essential. In this paper three different techniques based on grid index, modified to handle geo-textual queries using hybrid index, YPK-CNN, SEA-CNN and CPM to handle real time queries, are presented. The methods to find kNN based on these three techniques are proposed in this paper and are also compared. Conceptual partitioning along with hybrid index improve the query performance by 30 to 40%.

Keywords: Conceptual Partitioning · Geo-Textual data · Grid Index · Hybrid Index · Inverted List · Moving objects · Nearest Neighbour

1 Introduction

Geo-Textual data is generated abundantly for various objects in our surrounding due to the increase in the wide variety of GPS tracking handheld devices. Each object has location data (spatial) and attribute data (non-spatial) attached to it. Spatial data are 'where things are' data and attribute data are 'what things are'. The movement of these objects can be tracked and can be used to know the current location of the objects. The dynamic objects can be queried and are retrieved based on their location at the query time as well as the query result set can be updated based on the movement of the objects over time period. In case the query point is moving, the result set needs to be updated based on the current location of the query over the time period. This kind of scenario occurs in various applications like emergency management system, traffic management etc.

R. Manza et al. (Eds.): ACVAIT 2022, AISR 176, pp. 230–249, 2023.
https://doi.org/10.2991/978-94-6463-196-8_19

In real-time applications, maintaining location of objects synchronized with time is essential. When the velocity vector of the objects is not known in advance, capturing the current location of the objects and maintaining the results of the queries require efficient indexing techniques and search algorithms. In real time decision support systems, with multiple queries running simultaneously, we need to optimize the query processing time.

2 Related Work

To answer continuous range queries over collections of moving objects, the researchers have used grid index for query indexing and have proved that grid indexing yields better performance than other index structures like R-Trees [1]. A query management technique called MQM (Monitoring Query Management) is used for monitoring real time range-monitoring queries. An object reports its location to the server whenever its movement affects any query results (i.e., crossing any query boundaries) or it moves out of its resident domain [2]. Query Indexing and Velocity constraint indexing are demonstrated for continuous queries on moving objects using R-Tree and safe region, wherein objects report their new locations to the server periodically or when they have moved by a significant distance [3]. MobiEyes answers moving range queries over moving objects using R-Tree with assumption that queries move linearly with fixed velocity and monitors region of moving query [4].

The queries on moving objects can be classified as past (historical), present and future (predicate) queries [5]. The proposed method based on Query Indexing using R-trees answers present queries [3]. YPK-CNN, SEA-CNN and CPM are the solutions given for present queries based on the grid index [6,–8]. When objects' moving pattern of motion is not known in advance, they move in an unrestricted manner. Hence capturing the moved location of the objects and answering the updated query results occurs with some time delay. Techniques are proposed to reduce the time delay in present queries but textual attributes of the objects are not considered [6]–[8]. YPK-CNN re-evaluates an existing kNN query every T time unit and it makes use of its previous result in order to restrict the search space [6]. Shared Execution Algorithm-Continuous Nearest Neighbour Monitoring (SEA-CNN) techniques focuses on monitoring of NN with bookkeeping for answer region [7]. Conceptual Partitioning Monitoring (CPM) techniques does NN-Monitoring with Bookkeeping [8].D-Grid, dual space grid index for moving objects, indexes moving objects using both location and velocity spaces for answering range and kNN queries [9]. In which, authors have proposed D-Grid, an in-memory dual space grid index for moving objects, which indexes moving objects using grid structures in both location and velocity spaces [9].

Scalable INcremental hash-based Algorithm (SINA) implements R-Tree for answering static moving range queries. It uses shared execution and incremental evaluation technique to compute only the updates of the previously reported answers [10]. DISC (aDaptive Indexing on Streams by space-filling Curves) proposes B-tree that uses a space-filling curve to answer static and moving approximate kNN query [11]. Influential Neighbour Sets technique implements R*-tree and VoR-tree to answer moving kNN query by safeguarding objects [12]. In Data Frame Based Spatiotemporal Indexing, spatiotemporal trajectory retrieval algorithm is used to answer moving objects range query [13].

If the trajectories of the moving objects are known in advance, the predictive queries or future queries can be answered at the query processing time. The motion pattern of the objects decides the accuracy of the response to the predictive query. Influential Time Parameterised Indexing, Voronoi cells, updating previously reported answers and approximate nearest neighbour (NN) queries are some techniques used for answering predicate queries. Neighbour Sets are used instead of safe regions to improve performance [10,–12, 14–23]. The time-aware queries considers valid time of the objects and answers Boolean spatial keywords queries [24].

For spatial indexing, most of the techniques use R-tree. But in case of continuously moving objects, the nodes of the R-tree index need to split and merge to maintain the locations of continuously moving objects. This degrades the performance of maintaining R-Tree index structure and answering queries using this index structure. Hence Grid Index is preferred for maintaining locations of moving objects. In this paper the focus is on creating hybrid index structure using grid index and inverted list for spatial and textual components respectively. In case the moving objects movement pattern is not known in advance, the hybrid index helps in answering the geo-textual queries over these objects. The search algorithm based on grid index and inverted list hybrid index, prunes the search space and hence reduces the time delay for answering the nearest neighbours query.

The geo-textual indices combine spatial and text indexing techniques either tightly or loosely [25]. In text-first loose combination, postings in each inverted list is arranged using spatial structure whereas in spatial-first loose combination leaf nodes of spatial index contain inverted files or bitmaps for the textual information of the objects belonging to the region. On the contrary the tight combinations combine both the spatial and text indexes tightly helping to prune search space in an effective manner. R-tree, a grid or a space filling curve are commonly used spatial index structures and inverted files or bitmaps are used commonly for handling text index. In case of tight combinations, one technique integrates a text summary into every node of a spatial index, and another integrates the spatial information into each inverted list [25]. Authors of this paper have implemented a tight combination of hybrid index with text summary into every cell of grid index [5] to handle static and moving objects.

When both objects and queries are static, results do not change over time. When query points and/or objects move the query result changes as some kNN objects go out of scope and new objects come nearer to the query point. Keeping track of data objects moving in and out of the current kNN result set is essential to re-evaluate the queries.

Various hybrid index structures are proposed by researchers and they differ in structures and their effectiveness in answering queries. The indexing structure decides performance of query evaluation depending upon the pruning of the data space. Other factors affecting the query performance include the node splitting technique, search space like Euclidean or road networks. The demand for various kinds of queries like moving queries, approximate query, joint query, predicate query, group query, why not query has grown over the period.

3 Methodology

Geo-textual queries are classified based on the location of data objects with respect to time. If the data objects are queried based on the past locations of the moving objects, are called historical queries and are answered using the past trajectories of the objects. If the trajectories of the moving objects are fully predictable at query processing time, the future queries can be answered to know the position of the objects in future. Present queries are answered in real-time, where the future locations of the objects are not known in advance. Periodically the changes in locations of objects are noted and indexes are updated accordingly. This helps in updating the query results in real-time. In this paper three techniques viz. YPK-CNN, SEA-CNN and CPM, which use grid index for maintaining locations of moving objects, are modified to handle geo-textual queries and are presented [6–8].

Each geo-textual object has a point location defined by latitude and longitude at a particular time stamp and is also described by textual attributes. To answer geo-textual queries, we need to combine textual and spatial structures. While answering the geo-textual queries on moving objects and moving queries following issues need to be addressed with improved execution time.

- the index needs to be updated as the objects move
- affected queries need to be re-evaluated when any objects from its' answer set is moved
- moved query need to be re-evaluated
- queries need to be answered with short execution times for large numbers of moving objects and queries

In every processing cycle, index needs to be updated for all moved objects during the processing cycle, affected queries and moved queries need to be re-evaluated. The technique used for maintenance of result sets needs to optimize the time required for maintaining it.

In case of real-time queries i.e. present queries, the objects' motion patterns and velocity vectors are not known and hence their movements are non-predictable. The query points and data objects being dynamic in nature move frequently and arbitrarily. As the multiple queries are running simultaneously, continuous reporting of its updated results is essential. In this research, the Geo-Textual hybrid indexing techniques are used, with focus on Grid-Inverted List hybrid index, to minimize the index maintenance and query execution time.

Following 3 proposed algorithms are developed, implemented and studied to evaluate their performance against the original techniques:

1. Modified Hybrid Index and modified YPK-CNN search technique – Grid-Inverted List Hybrid Index structure is developed to implement modified YPK-CNN techniques to answer Geo-Textual queries on moving Geo-Textual objects.
2. Modified Hybrid Index and modified SEA-CNN search technique – Grid-Inverted List Hybrid Index structure is developed to implement modified SEA-CNN techniques to answer Geo-Textual queries on moving Geo-Textual objects.

3. Modified Hybrid Index and modified CPM search technique–Grid-Inverted List Hybrid Index structure is developed to implement modified CPM techniques to answer Geo-Textual queries on moving Geo-Textual objects.

In this implementation, tight combination is used where text and spatial indexes are combined tightly. Every cell of the grid-structure has inverted list of all the objects falling in cells' region.

3.1 Grid-Inverted List Hybrid Index

In grid-based index structure the region of interest is mapped to a 2-dimentional array of grid (Fig. 1). Each cell of the grid is mapped to some region of space. Let D(t) be a Geo-Textual data set of objects belonging to the region of interest at time t. Each spatial object o ∈ D(t) is defined as a pair (o.ρ, o.ϕ), where o.ρ is a 2-dimensional geographical point location (longitude, latitude) and o.ϕ is a text description of the object. The geographical location of an object o at time t is denoted as o(t).ρ = (o(t).long, o(t).lat). The grid with cells δ × δ, where δ is the number of cells in each direction, is used to create a Geo-Textual hybrid index. Each cell of the grid is uniquely identified by c[i, j], where 0 < = i, j < δ, i denotes the i^{th} cell (column) in horizontal direction and j denotes the j^{th} cell (row) in vertical direction starting from lower-left corner of the grid. The data objects are mapped to grid cells by using the Eqs. 1 and 2.

$$i = (o(t).long - minlong) * \delta / maxlong - minlong) \tag{1}$$

$$j = (o(t).lat - minlat) * \delta / maxlat - minlat) \tag{2}$$

In the above formulae, minlong and minlat are the minimum longitude and latitude respectively of the Region of Interest. Similarly, maxlong and maxlat are the maximum longitude and latitude respectively of the Region of Interest (Fig. 2).

In each cell c[i, j], an object list is created for all the objects (o(t)) mapped to it and also an inverted list for all these objects' text description is created and maintained in this cell as shown in Fig. 3.

Grid-Inverted List hybrid index tightly combines an inverted list to each cell of the grid index for all objects belonging to the region of interest. Every grid cell maintains the list of objects mapped to the cell and also the inverted list of textual attributes of the objects belonging to that cell. Figure 3 illustrates hybrid index structure in which each grid cell has an object list and inverted list. This tight combination assists in retrieving objects which fulfil text criteria within the vicinity of the query object by pruning the search space efficiently as it does not have to check all the objects for responding the query. Figure 4 demonstrates hybrid index structure for region of interest shown in Fig. 1.

3.2 KNN Query

Let D(t) be a geo-textual data set of objects belonging to region of interest at time t. Each spatial object $o \in D(t)$ is defined as a pair $(o.\rho, o.\varphi)$, where $o.\rho$ is a 2-dimensional geographical point location (latitude and longitude) and $o.\varphi$ is a text description of the

object [26]. The kNN query $q = (\varphi, \rho, k)$ takes three arguments, where $q.\varphi$ is a set of keywords, $q.\rho$ is a spatial query point, and $q.k$ is the number of objects to retrieve. The result of kNN, q(D(t)), is a set of k objects, each of which covers all the keywords in $q.\varphi$ [26]. Objects are ranked according to their distances to $q.\rho$ at time t. It can be represented mathematically by following relation,

$$\forall o \in q(D(t))((\nexists o\prime \in D(t) \backslash q(D(t)))(dist(o\prime.\rho, q.\rho) \leq dist(o.\rho, q.\rho)) \wedge q.\varphi \subseteq o\prime.\varphi) \tag{3}$$

The performance of creating the Grid-Inverted List Hybrid Index structure for the dataset D(t) and the geo-textual kNN query over this hybrid index structure is studied with three different techniques for index update and search algorithm.

3.3 Hybrid Index Implementation with YPK-CNN Technique

Xiaohui Yu et al. (2005) proposed a method called YPK-CNN for continuous monitoring of kNN queries. In this research work, this technique is modified to handle Geo-Textual queries using Geo-Textual Hybrid Index. In this technique objects are indexed with a grid of cells with size $\delta \times \delta$. Updates are not processed as they arrive and changes are directly applied to the grid. Every kNN query is re-evaluated every T time unit. When a query is evaluated for the first time for its kNN, two step search retrieves the result. In the first step, the cells around the query cell are traversed until k objects are found. In the second step, the search region is set as square with side length 2.d + e, where d is the distance of the farthest object from the query point found in the first step and e is the length of the cell (extent) in the grid index. All the cells intersecting search region are traversed to find the objects satisfying query criteria and the first k objects in the order of the distance from the query point are retrieved as the result.

To re-evaluate the existing query q, YPK-CNN uses its previous result set to restrict the search space. Farthest distance that the current kNN have moved is found and is denoted as d_{max}. The new search region is then the square centred at q with side length $2 \cdot d_{max}$ + e. Then all the cells intersecting new search region are traversed to find the objects satisfying query criteria. When query q changes its location, it is evaluated as a new query.

Figure 3 shows the Grid-Inverted list hybrid Index structure implemented for YPK-CNN. As shown in the figure it is a tight implementation of Geo-Textual Hybrid Indexing technique. Every cell has the object list belonging to the cell and Inverted list of keywords of all the objects mapped to the cell. Figure 5 describes the structure of the Query table for YPK-CNN implementation. It maintains the list of all current queries along with the query criteria and current result set.

In case of non-hybrid implementation, all the objects in the search region are evaluated for distance and text criteria separately. This results in false positive hits. In case of hybrid index text criteria of the query is checked with the inverted list of the cell. If any objects are found in the cell satisfying the text criteria, only those objects are checked for the distance from the query point. This pruning in the search space results in reducing the number of actual hits and hence false hits are avoided by using hybrid index structure.

3.4 Hybrid Index Implementation with SEA-CNN Technique

Xiaopeng Xiong et al. (2005) proposed a method called Shared Execution Algorithm – Continuous Nearest Monitoring (SEA-CNN) for continuous monitoring of kNN queries. This technique is modified to handle Geo-Textual queries using Geo-Textual Hybrid Index. In this technique also objects are indexed with a grid of cells with size δ X δ. When a query is evaluated for the first time for its kNN, two step search retrieves the result. In the first step, the cells around the query cell are traversed until k objects are found. In the second step, the search region is set as a circle with diameter 2·d + e, where d is the distance of the farthest object from the query point found in the first step and e is the length of the cell (extent) in the grid index. All the cells intersecting search region are traversed to find the objects satisfying query criteria and the first k objects in the order of the distance from the query point are retrieved as the result.

The changes in the kNNs of the existing queries are monitored to re-evaluate the queries, if there are any movements in the search region of the query. The current answer region of the query is set as a circle with q's location as the centre of the circle and with radius as the distance of the current k^{th} NN from the query point. The cells which intersect the answer region of q keep this book keeping information. In the next update cycle, after time T, if any cells intersecting the answer region of the query have been affected due to movements of the objects, this query is re-evaluated by setting the search region based on the following criteria.

1. If some of the current knns of the query move within the answer region or some new objects enter the answer region, the search region of the query is set as the current answer region.
2. If any of the current kNNs move out of the answer region, the radius of the search region is set as the d_{max}, where d_{max} is the distance of the object from the query point, which has moved farthest from the query point.
3. If the query point has moved to a new location q', the centre of the search region is Set as Q', and radius is set as Dist(Q, Q') + k-Dist, where k-dist is the distance of the current k^{th} NN from the query point Q.

All the cells intersecting the updated search region are processed to find the updated kNN set of the query point.

Figure 3 shows the Grid-Inverted list hybrid Index structure implemented for SEA-CNN. As shown in the figure it is a tight implementation of Geo-Textual Hybrid Indexing technique. Every cell has the object list belonging to the cell and Inverted list of keywords of all the objects mapped to the cell. Figure 5 describes the structure of the Query table for SEA-CNN implementation. It maintains the list of all current queries along with the query criteria and current result set. As in the case of YPK-CNN, the hybrid implementation prunes the search space, reducing the number of actual hits and hence false hits are avoided.

3.5 Hybrid Index Implementation with CPM Technique

Mouratidis et al. (2005) proposed a method called conceptual partitioning monitoring (CPM) for continuous monitoring of kNN queries. This technique is modified to handle

Geo-Textual queries using Geo-Textual Hybrid Index. In this technique also, like the previous two techniques, objects are indexed with a grid of cells with size δ X δ. When a query is evaluated for the first time for its kNN, the cells/rectangles around the query cell are added progressively to heap in the ascending order of the distance from the query point. Rectangles from each direction act as boundary boxes. The cells from the heap are removed and checked for the objects satisfying the query criteria till the next cell is at the distance greater than the k-dist, where k-dist is the distance of the k^{th} object in the kNN set. All unexplored regions fall in some rectangle in some direction which is at a distance greater than the k-dist. CPM maintains a query table, wherein for each query it maintains its location coordinates, the k-dist, its current kNN result set, visit list and search heap. CPM is able to monitor the updates to objects and kNN result sets with this book keeping. CPM also maintains an influence list along with an object list in each cell of the grid index. The influence list maintains the list of queries whose influence region contains the cell i.e. the cell is at a distance less than equal to the k-dist of the query.

Figure 6 shows the Grid-Inverted list hybrid Index structure implemented for CPM. As shown in the figure it is a tight implementation of Geo-Textual Hybrid Indexing technique. Every cell has the object list belonging to the cell and Inverted list of keywords of all the objects mapped to the cell. Every grid cell also has the influence list of queries, specifying to which query influence region this cell belongs. It is used for book-keeping. Whenever any object from this cell moves out or any objects enter in the cell, the queries from influence lists are checked if they are affected and are accordingly re-evaluated in each update cycle. Figure 7 describes the structure of the Query table for CPM implementation. It maintains the list of all current queries along with the query criteria, current result set, heap and visit list of each query. Heap and visit list are used while re-evaluating the query and saving the time of creating heap and calculating distance of surrounding cells from the query point. When the query is re-evaluated, cells in the visit list are traversed in the order to find the updated kNN set. If the visit list is exhausted and kNN are not found only then the heap is processed for further search.

As in the case of YPK-CNN and SEA-CNN, the hybrid implementation of Grid Index and Inverted List, helps in pruning the search space, reducing the number of actual hits and hence false hits are avoided. This helps in improving the performance of geo-textual query.

3.6 Differences Between YPK-CNN, SEA-CNN and CPM Techniques

Book keeping is done by the three algorithms to evaluate the kNN query in each update cycle i.e. at every timestamp. Figure 8 shows the difference between the three methods pictorially. As can be seen in the figure, the number of grid cells accessed are varied. YPK-CNN computes the search region bounded by square, around the query point and then accesses the cells intersecting with the search region. SEA-CNN computes the search region bounded by the circle, the query point and then accesses the cells intersecting with the search region. Whereas CPM works with conceptual partitioning and visits the next cell only if required. As can be seen from the figure more number of cells are visited using YPK-CNN technique as compared to SEA-CNN technique and more number of cells are visited in case of SEA-CNN technique as compared to CPM technique. These differences among three methods decide the number of cells visited

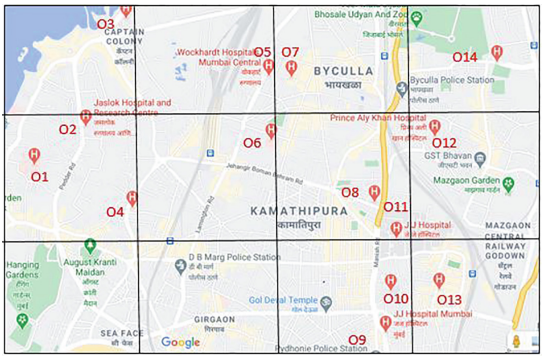

Fig. 1. Region of Interest with Grid

and hence the number of objects evaluated. The hybrid index implementation narrows down the search further by visiting a cell only if it satisfies the query criteria and further objects are evaluated only if they satisfy the query criteria.

In case of modified YPK-CNN and modified SEA-CNN, the speed of the object decides the area of the updated search region, as the farther the object moves, the area of the search region increases. Thus the performance of the modified YPK-CNN and modified SEA-CNN algorithms depends on the speed of the objects whereas the modified CPM is not affected by the speed as it searches for objects only in the influence region of the query.

In case of modified YPK-CNN and the modified SEA-CNN techniques, redundant and more cells are processed as compared to the modified CPM technique which uses conceptual partitioning to restrict the search space. The modified CPM also does the bookkeeping of objects moving inside and outside the influence region in every update cycle. This helps in reducing unnecessary computations. When the query point is moved, the modified YPK-CNN and the modified CPM techniques evaluate the query result like a new query. But SEA-CNN technique uses a formula to find the updated search region to re-evaluate the query. SEA-CNN assumes that the only updates are from incoming objects and/or NNs that move within the distance of k^{th} object from q. It performs redundant computations in several cases.

3.7 Proposed Algorithms

In the beginning, all the objects in the data set are scanned for spatial and non-spatial attributes and the Grid-Inverted List hybrid index is built for the data set. Then all the initial queries are evaluated using the created hybrid index structure. After this periodically, in every update cycle all the movements of the objects and queries are captured and the index is updated with the new locations of the moved objects. Then all the moved queries, new queries and affected queries are evaluated based on the updated index,

Following algorithms are developed for each of the techniques namely, YPK-CNN Hybrid, SEA-CNN Hybrid, CPM Hybrid.

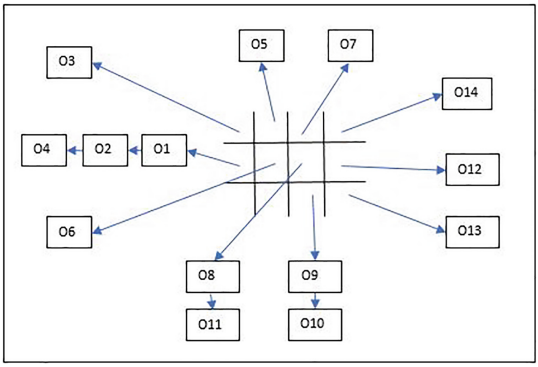

Fig. 2. Grid Index for the region of interest

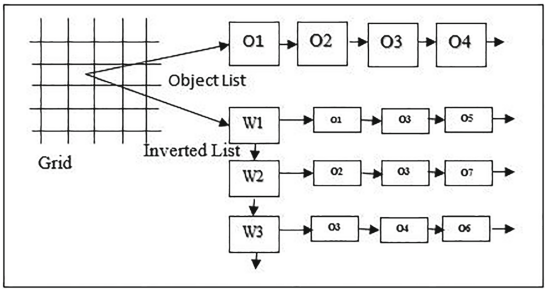

Fig. 3. Grid and Inverted List Hybrid Index

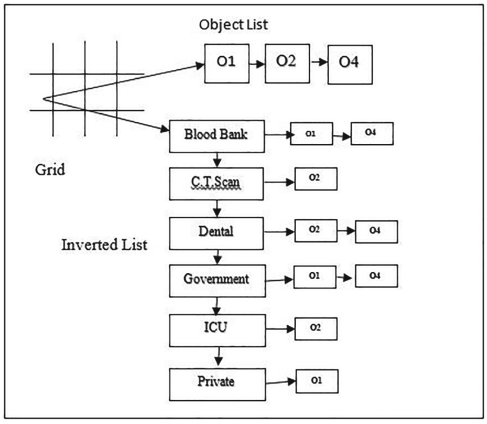

Fig. 4. Grid and inverted Hybrid Index

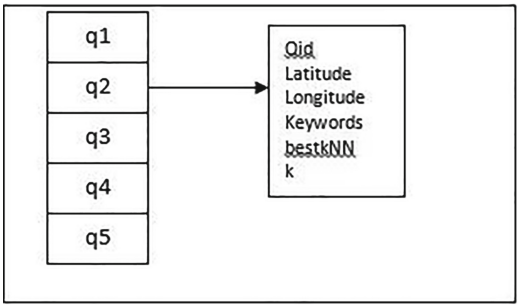

Fig. 5. Query Table structure for YPK-CNN and SEA-CNN techniques

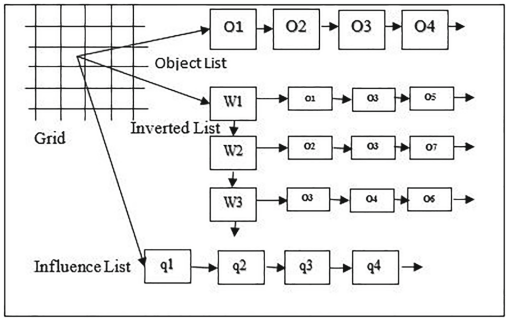

Fig. 6. Grid-Inverted List Hybrid Index Structure for CPM

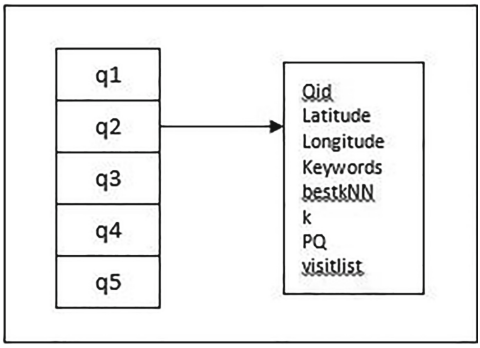

Fig. 7. Structure of Query table for CPM

3.7.1 Hybrid Index Construction Algorithm

This algorithm constructs the hybrid Index for the geo-textual data objects when the grid size m X n and bounding region of interest (spatial coordinates) are provided to the algorithm. It first creates the grid of the provided size and then finds the cell c[i j] for each object in the database using the Eqs. 1 and 2 and then adds this object to the object

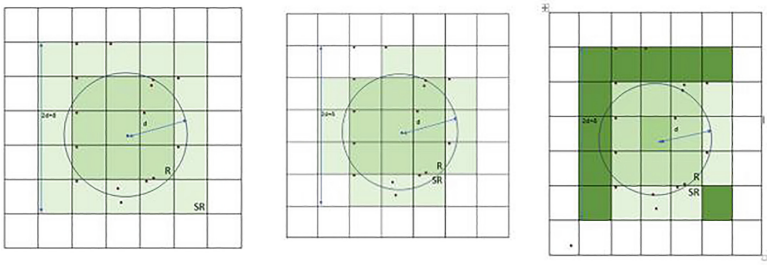

Fig. 8. differences between YPK-CNN, SEA-CNN and CPM techniques

list of the corresponding grid cell. Then the inverted list of the cell is updated for the object added to it. This algorithm creates a tight grid-inverted list hybrid index.

3.7.2 Hybrid Index Updation Algorithm

In every update cycle, all the changed locations of the geo-textual objects and queries are captured. If the object is moved in the same cell, the algorithm updates the latitude and longitude of the objects. If the object has moved to a different cell or it is terminated, the algorithm removes the object from the object list of the old cell and also removes its entries from the inverted list of the old cell. If the object has moved to different cell or new object has entered the region of interest, the algorithm finds the new cell c[i j] using the Eqs. 1 and 2 and then adds this object to the object list of the corresponding grid cell and also updates the inverted list of the cell for the object added to it.

3.7.3 KNN Search Algorithms for Modified YPK-CNN

This algorithm evaluates geo-textual kNN query at time t using modified YPK-CNN techniques. It first initializes the kNN set of the query to NULL. Then it finds a query cell using the Eqs. 1 and 2. After initialization, starting with the query cell and its surrounding cells in each direction, it checks for the objects in the cells till k objects are found satisfying query criteria. After the k objects are found, the search region area is initialized as square with side 2 * k-dist + e and query q at the centre. All the cells interesting this search region are then processed to find all the objects satisfying query criteria and the k nearest neighbours of the query point.

3.7.4 KNN Search Algorithm for Modified SEA-CNN

This algorithm evaluates geo-textual kNN query at time t using modified SEA-CNN techniques. It first initializes the kNN set of the query to NULL.Then it finds a query cell using the Eqs. 1 and 2. After initialization, starting with the query cell and its surrounding cells in each direction it checks for the objects till k objects are found satisfying query criteria. Then the search region area is initialized as a circle with diameter 2 * k-dist + e and query point q as the centre. All the cells interesting this search region are then processed to find all the objects satisfying query criteria and the k nearest neighbours of the query point.

3.7.5 KNN Search Algorithm for Modified CPM

This algorithm evaluates geo-textual kNN query at time t using modified CPM techniques. It first calculates the query cell c[i, j] using the Eqs. 1 and 2. First few entries to the heap consist of the query cell and the bounding rectangles to the query cell in each direction. These entries in the heap are in ascending order of key value as the minimum distance between query point and the bounding rectangles. Then in iteration, entries in the heap are removed one by one and are processed till kNN are found or the heap is empty. If the removed entry from the heap is a cell, then its inverted list is checked to see if it has an object satisfying query criteria. If a cell has such objects then these are processed to check if they are kNN of the query, if so kNN and k-dist are updated. Removed entry is added to the visit list of the query and then this query is added to the influence list of the cell. If the removed entry is a rectangle, then each cell of the rectangle is added to the heap and the next level bounding rectangle in the same direction is added to the heap. This algorithm returns the kNN set of the query q as output.

3.7.6 Update Handling Algorithm for Modified YPK-CNN

Update handling of affected queries due to moved geo-textual objects and queries is done in this algorithm using modified YPK-CNN technique. If the query point has moved, it is evaluated as a new query. Otherwise if some kNN objects have moved outside the answer region, new Search Region with formula 2.d + e, is set based on the farthest distance kNNs have travel (d) and grid cell extent e else search region is set as the answer region and the query is re-evaluated.

3.7.7 Update Handling Algorithm for Modified SEA-CNN

If some kNN objects have moved outside the answer region, a new search region with formula 2.d + e, is set based on the farthest moved kNN (d) and grid cell extent e else search region is set as the answer region and the query is re-evaluated. If the query point has moved, a new search region is set based on the movement of the query and farthest moved kNN and query is re-evaluated.

3.7.8 Update Search Algorithm for Modified CPM

This algorithm re-evaluates geo-textual kNN query at time t using modified CPM techniques. It first initializes k-dist to infinity and initializes list kNN to NULL Then for each cell in the visit list of the query, it checks the inverted list to see if any objects exist in the cell satisfying query criteria. If a cell has objects satisfying query criteria, it processes these objects to find kNN. If kNN objects are not found and the visit list is exhausted, then it starts processing the entries in the heap till kNN are found satisfying query criteria.

3.7.9 Update Handling Algorithm for Modified CPM Technique

Update handling of affected queries due to moved geo-textual objects and queries is done in this algorithm using modified CPM technique. First of all, for each moved object, its

old cell's and new cell's influence region are checked for affected queries. If the moved object belongs to the kNN of the old cells' influence regions query and it has moved out of the answer region of the query, then out object count of the query is incremented. If this moved object has entered the answer region of the query belonging to the new cell's influence list, then in object count is incremented and object is added to in object list of the query. Once affected queries are marked, for each query from the query table is checked and processed. If the query is moved in the cell itself, search updates the result by processing the visit list. If the query is moved to a different cell its result is evaluated like a new query. If the query is static and sufficient objects are moved inside the answer region of the query, kNN list is updated with existing list and in object list else query is re-evaluated. If the query is static and kNN objects have moved in the answer region itself, the kNN list is updated with the existing list and values are updated.

4 Experimental work

Collecting real-time Geo-Textual data sets is tedious and it usually takes a substantial amount of time and effort. In this research, the python code is written to generate the initial data set and update cycle's data sets by varying parameters and by considering different types of movements of Geo-Textual data. Various data sets are generated using simple random (probability) sampling technique, where every element has an equal chance of getting selected to be the part sample. Various data sets used in experimental evaluation are generated by using 3 different speeds slow, medium and fast for moving objects and moving queries. Algorithms are evaluated by varying parameters like number of objects in a data set (N), number of queries (n), number of nearest neighbours (k), grid cell size (δ), fraction of the objects updates and query updates. Updates to the objects and queries include movements of objects/queries within the region of interest (movement), leaving of objects/queries from the region of interest (terminate) and new objects/queries entering the region of interest (new). Updates are captured and evaluated every update cycle. Each update cycle first captures the updates to location of objects and.

queries. It then updates the index structure for objects updates and then evaluates the affected and updated queries.

Algorithms presented in Sect. 3.7 for proposed solutions of optimization of Geo-Textual queries on Geo-Textual moving queries and objects are implemented in python. These algorithms are analysed using various data sets. Spyder, the Scientific Python

Table 1. Example Data Set

Data set	N	n	k	δ × δ	Number of Update cycles	Objects update count	Query update count
7	10000	100	1,2,3,4,5	5 × 5, 10 × 10, 15 × 15, 20 × 20, 25 × 25	3	121, 112, 115	12, 10, 11

Development Environment is used for python code implementation. To visualize the output of Geo-Textual objects, queries and query results, the Jupyter Notebook environment with ArcGIS API is used. In this environment, Geo-Textual experimental data is converted to ESRI shape file and then this shape file is imported as a feature layer on the map for visualization.

For performing experiments various generated data sets are used. One such example is shown in Table 1. Data set 7 consists of 10000 (N) Geo-Textual Objects and 100 (n) queries. This data set is evaluated for 1-NN, 2-NN, 3-NN, 4-NN, and 5-NN with grid sizes [5 X 5], [10 X 10], [15 X 15], [20 X 20], and [25 X 25]. Out of 10000 Geo-Textual objects 121 objects are updated in the first update cycle for their locations, 115 objects moved in the second update cycle and 1441 objects moved in the third update cycle. Out of 100 queries, 12 are moved in the first update cycle, 10 moved in the second update cycle and 11 moved in the third update cycle. In this example the set is processed for 5 kNN values and 5 grid sizes giving rise to 25 combinations.

For each dataset all three techniques namely, modified YPK-CNN hybrid, modified SEA-CNN hybrid and modified CPM hybrid are processed. For each of these techniques, the number of cells visited in the influence region, number of inverted lists accessed, number of cells visited and number of objects processed for kNN computation and re-evaluation of results in each update cycle are measured. On these measured values for each execution, the total CPU performance cost is calculated. The results of the analysis performed are presented in Sect. 5.

5 Results and Discussion

The Geo-Textual hybrid indexing techniques and search algorithm based on these indexing techniques for processing moving objects and queries are implemented and analysed for 3 proposed techniques. Number of Geo-Textual objects, number of Geo-Textual queries, grid size, number of location updates for objects and number of location updates for queries are the parameters evaluated for performance analysis of the algorithms.

Figure 9 shows the actual number of objects evaluated and the number of objects satisfying the query criteria in the search region in case of non-hybrid index. In case of hybrid index, numbers of actual hits are reduced and hence false hits are avoided by using inverted index in hybrid index structure thus pruning the search space.

Varying Grid Size δ. In Fig. 10, the grid size is changed by keeping other parameter values constant. For smaller values of δ, the number of cells in the grid is less, resulting in longer object lists and inverted lists with more number of nodes to be visited. For larger δ, the number of cells in the grid is increased. The time taken to traverse the list is decreased due to the distribution of objects in more number of cells. But when the numbers of cells are increased, there is high overhead due to heap operations. The time of NN computation for a new or a moving query depends strongly on the cell size; a large value for δ incurs high overhead due to heap operations, while a small value implies a high number of objects processed in the influence region.

Varying Value of k. Figure 11 shows the effect of changing the value of k, number of nearest neighbours, on evaluating queries by keeping other parameter values constant.

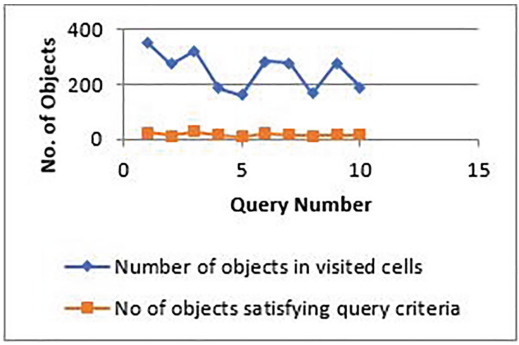

Fig. 9. Number of Objects in visited cells and number of objects satisfying query criteria in Non-Hybrid Index

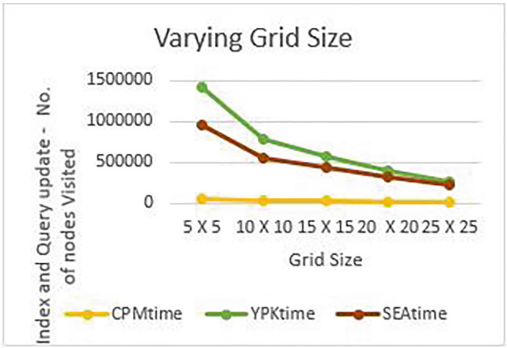

Fig. 10. SEQ Figure * ARABIC Grid Size Vs performance

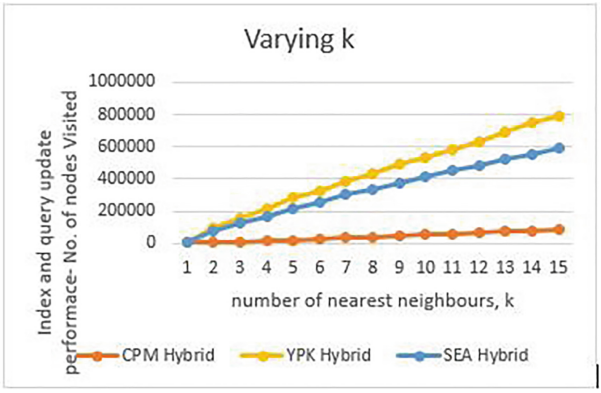

Fig. 11. Number of nearest neighbours (k) vs performance

As the value of k is increased, the time taken to update the result of kNN query is more. To find the k nearest neighbours more cells are visited as the value of k is increased. As the graph illustrates, modified CPM Hybrid Index gives better performance among the 3 algorithms.

Varying N – Index and Query Update. Figure 12 demonstrates index and query update performance dependency on the value of N. The time taken to update indexes and answer queries is directly proportional to the number of objects. Computation cost of the algorithm depends upon N.. Index update time is linear to N and number of updates to objects and increase in object movements does not affect the performance.

Varying n. As shown in Fig. 13, as the number of queries increases, the query result maintenance time is increased. As the number of updates to the query grows, the time required will also be more.

Figure 14 shows example 3NN query with 5 × 5 grid implemented and visualised in Jupyter Notebook.

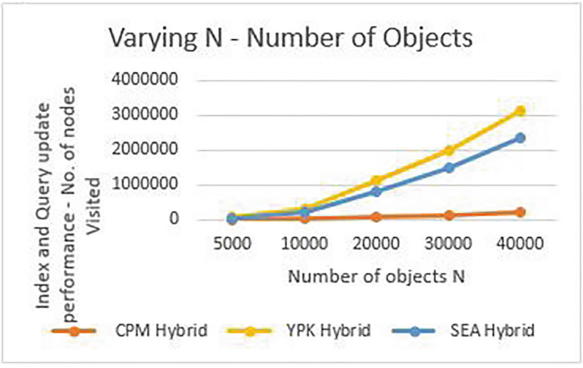

Fig. 12. Number of objects (N) vs Performance

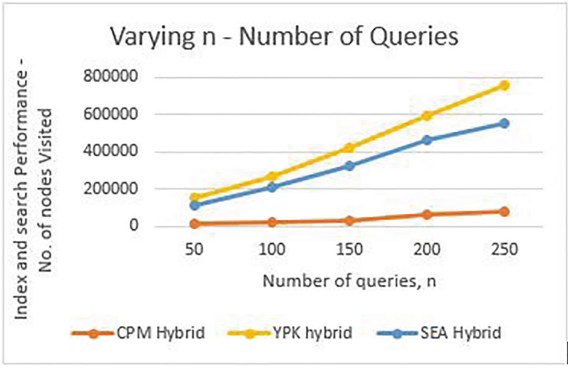

Fig. 13. number of queries (n) vs performance

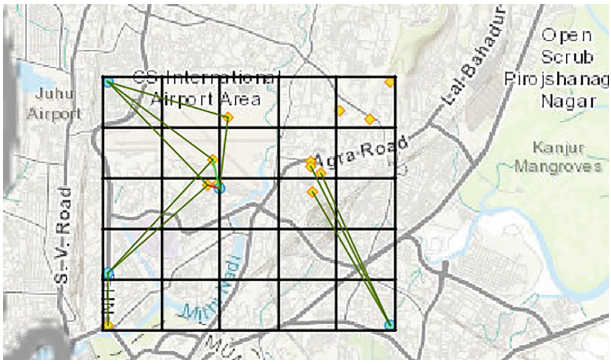

Fig. 14. 3NN query 5 × 5 grid implemented in Jupyter Notebook

6 Conclusion

This paper presents the comparative study of 3 techniques modified to handle geo-textual moving objects with unknown movement patterns using grid-inverted list hybrid index. Uniform grid makes it easy to calculate the cell of a given object with its latitude and longitude in constant time. The Grid index is a space-driven or data-independent method and partitions the data space in a way that reflects the data distribution in a given data set. Hence maintaining grid-index for moving objects can be conveniently by just removing objects from one cell and adding to another. The grid index guarantees that any point query can be answered in, at most, two disk accesses depending on whether the grid file is stored in main memory or disk.

Hybrid indexing having grid and inverted lists tightly combined, the search space narrows down. It reduces the number of false positive hits to database records Conceptual partitioning aids in processing the objects in the vicinity of the query point thereby restricting the search space in kNN processing. NN monitoring using bookkeeping information avoids unnecessary processing of objects and further improves the query performance. Hybrid index along with conceptual partitioning improves the query performance by 30 to 40%.

The extensive experiments demonstrate reducing the update and search cost for uniform Grid-Inverted List Hybrid Index. Future work can further improve the cost by implementing the multilevel Grid Index for skewed data. Euclidean space is considered in the search algorithms implemented in this research, extendibility of it needs to be checked for road networks. kNN search query algorithms are implemented in this research, and other types of Geo-Textual queries on hybrid index are in the further research scope. Keyword queries can further be studied for natural language processing (NLP) queries, fuzzy queries and other variations.

Acknowledgments. The author would like to thank Dr. Babasaheb Ambedkar Marathwada University, Aurangabad (MS) India for providing the publication support.

References

1. D. V. Kalashnikov, S. Prabhakar, and S. E. Hambrusch, "Main Memory Evaluation of Monitoring Queries Over Moving Objects," Distributed and Parallel Databases, vol. 15, no. 2, pp. 117–135, Mar. 2004, doi: https://doi.org/10.1023/B:DAPD.0000013068.25976.88.
2. Ying Cai, K. A. Hua, and Guohong Cao, "Processing range-monitoring queries on heterogeneous mobile objects," 2004, pp. 27–38. https://doi.org/10.1109/MDM.2004.1263040.
3. S. Prabhakar, Yuni Xia, D. V. Kalashnikov, W. G. Aref, and S. E. Hambrusch, "Query indexing and velocity constrained indexing: scalable techniques for continuous queries on moving objects," IEEE Transactions on Computers, vol. 51, no. 10, pp. 1124–1140, Oct. 2002, https://doi.org/10.1109/TC.2002.1039840.
4. B. Gedik and L. Liu, "MobiEyes: Distributed Processing of Continuously Moving Queries on Moving Objects in a Mobile System," in Advances in Database Technology - EDBT 2004, vol. 2992, E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, and E. Ferrari, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 67–87. https://doi.org/10.1007/978-3-540-24741-8_6
5. Sulbha Powar and Dr. Ganesh Magar, "Efficient Geo-Textual Hybrid Indexing Techniques for Moving Objects and Queries," IJRTE, vol. 8, no. 6, pp. 4419–4428, Mar. 2020, doi: https://doi.org/10.35940/ijrte.F9142.038620.
6. Xiaohui Yu, K. Q. Pu, and N. Koudas, "Monitoring k-Nearest Neighbor Queries over Moving Objects," 2005, pp. 631–642. https://doi.org/10.1109/ICDE.2005.92.
7. Xiaopeng Xiong, M. F. Mokbel, and W. G. Aref, "SEA-CNN: Scalable Processing of Continuous K-Nearest Neighbor Queries in Spatio-temporal Databases," 2005, pp. 643–654. https://doi.org/10.1109/ICDE.2005.128.
8. K. Mouratidis, D. Papadias, and M. Hadjieleftheriou, "Conceptual partitioning: an efficient method for continuous nearest neighbor monitoring," 2005, p. 634. https://doi.org/10.1145/1066157.1066230.
9. X. Xu, L. Xiong, and V. Sunderam, "D-Grid: An In-Memory Dual Space Grid Index for Moving Object Databases," in 2016 17th IEEE International Conference on Mobile Data Management (MDM), Porto, Jun. 2016, pp. 252–261. https://doi.org/10.1109/MDM.2016.46.
10. M. F. Mokbel, X. Xiong, and W. G. Aref, "SINA: scalable incremental processing of continuous queries in spatio-temporal databases," 2004, p. 623. https://doi.org/10.1145/1007568.1007638.
11. N. Koudas, T. Labs-Research, B. C. Ooi, K.-L. Tan, and R. Zhang, "Approximate NN Queries on Streams with Guaranteed Error/performance Bounds," p. 12, 2004.
12. C. Li, Y. Gu, J. Qi, G. Yu, R. Zhang, and W. Yi, "Processing moving k NN queries using influential neighbor sets," Proceedings of the VLDB Endowment, vol. 8, no. 2, pp. 113–124, Oct. 2014, doi: https://doi.org/10.14778/2735471.2735473.
13. C. Lv, Y. Xu, J. Song, and P. Lv, "A data frame based spatiotemporal indexing algorithm for moving objects," in 2016 12th World Congress on Intelligent Control and Automation (WCICA), Guilin, China, Jun. 2016, pp. 2592–2597. https://doi.org/10.1109/WCICA.2016.7578643.
14. Simonas ˇSaltenisy Christian S. Jenseny Scott T. Leuteneggerz Mario A. Lopezz, "Indexing the Positions of Continuously Moving Objects." ACM, 2000.
15. R. Benetis, C. S. Jensen, G. Karciauskas, and S. Saltenis, "Nearest neighbor and reverse nearest neighbor queries for moving objects," 2002, pp. 44–53. https://doi.org/10.1109/IDEAS.2002.1029655.
16. C. Shahabi, "Time Parameterized Queries in Spatio Temporal Databases," p. 31.
17. Y. Tao, D. Papadias, and J. Sun, "The TPR*-Tree: An Optimized Spatio-Temporal Access Method for Predictive Queries," p. 12, 2003.

18. K. Raptopoulou, A. N. Papadopoulos, and Y. Manolopoulos, “Fast Nearest-Neighbor Query Processing in Moving-Object Databases,” p. 25, 2003.
19. G. S. Iwerks, H. Samet, and K. P. Smith, “Maintenance of K -nn and spatial join queries on continuously moving points,” ACM Transactions on Database Systems, vol. 31, no. 2, pp. 485–536, Jun. 2006, doi: https://doi.org/10.1145/1138394.1138396.
20. An Chunming and Li Zongsen, “Studies on kNN query of moving objects for location management in spatial database,” Dec. 2013, pp. 2428–2432. https://doi.org/10.1109/MEC.2013.6885443.
21. Y.-K. Huang, Z.-H. He, C. Lee, and W.-H. Kuo, “Continuous Possible K-Nearest Skyline Query in Euclidean Spaces,” Dec. 2013, pp. 174–181. https://doi.org/10.1109/ICPADS.2013.35.
22. Y. Park et al., “A New Spatial Index Structure for Efficient Query Processing in Location Based Services,” 2010, pp. 434–441. https://doi.org/10.1109/SUTC.2010.64.
23. C. Li, Y. Gu, J. Qi, G. Yu, R. Zhang, and Q. Deng, “INSQ: An influential neighbor set based moving kNN query processing system,” May 2016, pp. 1338–1341. https://doi.org/10.1109/ICDE.2016.7498339.
24. G. Chen, J. Zhao, Y. Gao, L. Chen, and R. Chen, “Time-Aware Boolean Spatial Keyword Queries,” IEEE Trans. Knowl. Data Eng., vol. 29, no. 11, pp. 2601–2614, Nov. 2017, doi: https://doi.org/10.1109/TKDE.2017.2742956.
25. L. Chen, G. Cong, C. S. Jensen, and D. Wu, “Spatial Keyword Query Processing: An Experimental Evaluation,” p. 12, 2013.
26. S. K. Powar and D. G. M. Magar, “Spatial Keyword Query Processing: R*-IF Tree Implementation", Feb 18, pp 65–72, ISSN 2321–3469