International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-6, March 2020

Efficient Geo-Textual Hybrid Indexing Techniques for Moving Objects and Queries

Sulbha Kishor Powar, Ganesh Magar

Abstract -Advancements of various Geographic Information Technologies have resulted in huge growth in Geo-Textual data. Many Indexing and searching algorithms are developed to handle this Geo-Textual data which contains spatial, textual and temporal information. In past, Indexing and searching algorithms are developed for the applications in which the object trajectory or velocity vector is known in advance and hence we can predict the future position of the objects. There are real time applications like emergency management systems, traffic monitoring, where the objects movements are unpredictable and hence future position of the objects cannot be predicted. Techniques are required to answer the geo-textual kNN query where the velocity vectors or trajectories of moving and moving queries are not known. In case of moving objects, capturing current position of the object and maintaining spatial index optimally is very much essential. The hybrid indexing techniques used earlier are based on R-tree spatial index. The nodes of the R-tree index structure are split or merged to maintain the locations of continuously moving objects, increasing the maintenance cost as compared to the grid index. In this paper a solution is proposed for creating and maintaining hybrid index for moving objects and queries based on grid and inverted list hybrid indexing techniques. The method is also proposed for finding Geo-Textual nearest neighbours for static and moving queries using hybrid index and conceptual partitioning of the grid. The overall gain reported by the experimental work using hybrid index over the non- hybrid index is 30 to 40 percent depending on the grid size chosen for mapping the data space and on the parameters of queries.

Keywords: Conceptual Partitioning, Grid, Hybrid Index, Inverted List, Nearest Neighbour

I. INTRODUCTION

Geo-Textual data is generated on every click of time with wide availability of various techniques for knowing locations of objects and with various mobile objects. The movement of the dynamic objects can be tracked with time and dynamic or static objects can query over the other such objects. Table 1 shows the 4 possibilities of queries with dynamic nature of the object and queries..

Manuscript received on March 15, 2020.
Revised Manuscript received on March 24, 2020.
Manuscript published on March 30, 2020.
* Correspondence Author
Sulbha Kishor Powar, Research Scholar, Department of Computer Science, Women's University, Santacruz (West), Mumbai, Maharashtra, India.
Dr. Ganesh M. Magar, Associate Professor, Department of Computer Science, Women's University, Santacruz (West), Mumbai, Maharashtra, India.

The queries need to be answered based on the current location of the query objects and other moving objects and also matching query attributes to the attribute data attached to the objects This kind of scenarios occur in many decision making applications like transportation, agriculture, emergency management system, resource management system, marketing, traffic monitoring, security systems and many more.

Table 1 Type of objects and queries

Sr. No.	Object	Query	Description
1	Static	Static	Results do not change with time, Query Once
2	Static	Moving	Query result change over time as query point moves
3	Moving	Static	Affected Queries results change over time as knn object goes out of scope or new objects come nearer processing a continuous kNN query involves keeping track of data objects moving in and out of the kNN result set
4	Moving	Moving	Query result change over time as query point moves Affected Queries results change over time as knn object goes out of scope or new objects come nearer Involves monitoring the locations of query points and moving objects periodically

The continuous moving Geo-Textual objects and queries need to maintain their location attributes namely latitude and longitude with time. The dataset of these objects has combination of static and dynamic objects, which can query over each other. The dynamic nature of the data causes consistently reporting of results of the queries very much essential. In decision support system, multiple queries are processed simultaneously.

The objects and queries move dynamically and their velocity vectors and motion patterns are unknown and hence queries need to be answered in real time by minimizing query processing time.

II. RELATED WORK

The research is done on various techniques of indexing the data that contains both text descriptions and geo-locations. The indexing supports the efficient evaluation of geo-textual queries based on location and a set of keywords provided to the query and return the result set that matches the query criteria. To handle moving queries the indexing techniques need to maintain the changed location of objects and queries.

For creating geo-textual hybrid index, one needs to pick up a spatial index and a text index. Various techniques are developed for hybrid indexing which use inverted file and R-tree. Different combining schemes of hybrid indexing structure which integrate inverted files and R*-tree was proposed [1]. With Hybrid Index, the pruning power of both space and text is exploited simultaneously, thus merging the two steps into one and hence greatly enhancing the performance. Information retrieval R-tree (IR2-Tree) combines an R-tree with superimposed text signatures [2]. To efficiently answer top-k spatial keyword queries, the tight integration of data structures and algorithms is used with each node containing both spatial and keyword information. The IR-tree is studied and used extensively in variations of spatial keyword queries [3]. The framework proposed leverages the inverted file for text retrieval and the R-tree for spatial proximity querying[4].

Though much research is done on R-tree index, hybrid index created using other spatial indexing style are also proposed. Hybrid indices are implemented using a spatial indexing based on coarse-space partitioning (CSP) and a space-filling curve (SFC) [5]. Index structure that combines K-d tree and inverted file for spatial range keyword query was also proposed [6]. Grid spatial index were studied and proved that query times for tightly coupled indexing structures were faster than loosely coupled indexing structures[7]. Spatial-keyword inverted file (SKIF) and length-constrained maximum-sum region queries were developed using grid spatial index[8], [9]. Hybrid index, called inverted file quad-tree (IQ-tree) was proposed to answer Boolean range continuous queries (BRCQ) over a stream of incoming geo-textual objects in real time [10]. A scalable integrated inverted index, named I3 was proposed, which adopts the quadtree structure to hierarchically partition the data space into cells [11]. Top k spatial keyword search (TOPK-SK), and batch top k spatial keyword search (BTOPK-SK) were studied based on the inverted index and the linear quad tree, called inverted linear quadtree (IL-Quadtree)[12]. Grid-based signatures hybrid structure was proposed to handle regions-of-interest (ROIs) queries[13].

Most of the work done to handle Geo-Textual objects is either for answering dynamic queries on static objects or for answering static queries on dynamic objects [14], [15]. The proposed solutions answer the future queries based on the velocity vector of the moving objects or moving queries [16], [17]. Some researchers have worked on query indexing techniques to avoid maintaining the index in cases of dynamic objects [16], [18]. Few of these solutions consider the Geo-Textual hybrid indexing. The queries for dynamic objects are classified as past (historical), present and future (predicate) queries as shown in table 2. The solutions given for present queries are based on the grid index, but these do not consider the textual attributes of the objects [18]–[20].

Table 2 Types of queries depending upon the time of the query

Future queries (predictive queries)	The trajectories of the objects are fully predictable at query processing time	predicative or approximate query answering by making assumptions on the motion patterns of the objects
Present queries	The objects are highly Dynamic. Each object can move in an unrestricted fashion. i.e. we do not assume any pattern of motion	The objects' movements are non-predictable hence capturing the current locations of objects and queries the exact answers with a time delay (Δt)
Past queries (Historical queries)	The trajectories of the objects movements are captured and stored	Historical queries are answered based on the past trajectories of the objects

In case of predictive queries, the trajectories of the moving objects are known at the query processing time. The answer provided for predictive queries are approximated based on the motion patterns of the object. The techniques used for solving such queries include Time parameterised indexing, Voronoi cells, updating previously reported answers and approximate nearest neighbour (NN) queries. Influential Neighbour Sets are used instead of safe regions to improve performance [21]–[33]. Spatio-Temporal indexing based on data frames is developed which uses Geohash based spatial index [34]. D-Grid, dual space grid index for moving objects, indexes moving objects using both location and velocity spaces [35]. The time-aware queries considers valid time of the objects and answers Boolean spatial keywords queries [36].

When the objects and queries move in unrestricted manner, moving pattern of motion is not known in advance. In real time applications where the objects movements are not known in advance, the queries are answered with some time delay [18]–[20]. The techniques used for solving such type of queries focus on reducing the time delay with which the queries are answered. Table 3 highlights the indexing methods used for moving/static objects and moving/static queries.Most of the techniques used in past use R-tree for spatial indexing.

The nodes of the R-tree index structure are split or merged to maintain the locations of continuously moving objects. Much time is spent to maintain R-tree index structures for moving objects.

Table 3: Related Work

Method	Indexing technique	Query type	Key points
Query-index	Queries are indexed by R-tree	static range query moving objects	Safe Regions
Query-index	Queries are indexed by Grid	static range query moving objects	main memory evaluation
Monitoring Query Management	R-Tree	static Range query moving objects	Object's resident region
MobiEyes	R-Tree	moving range query moving objects	with assumption queries move linearly with fixed velocity monitoring region of a moving query
Scalable INcremental hash-based Algorithm (SINA)	R-Tree	static and moving range query	shared execution and incremental evaluation - computes only the updates of the previously reported answers
aDaptive Indexing on Streams by space-filling Curves (DISC)	B-tree that uses a space-filling curve	static and moving approximate kNN query	Approximate NN Queries on Data Streams
Xiaohui Yu, K. Q. Pu, and N. Koudas – Continuous Nearest Neighbour Monitoring (YPK-CNN)	Grid index	static and moving kNN query moving objects	re-evaluating an existing kNN query every T time units makes use of its previous result in order to restrict the search space
Shared Execution Algorithm - Continuous Nearest Neighbour Monitoring (SEA-CNN)	Grid-Index	static and moving kNN query moving objects	focuses on monitoring of NN with book keeping for answer region
Conceptual Partitioning Monitoring (CPM)	Grid Index	static and moving kNN query moving objects	Conceptual Partitioning NN-Monitoring with Book Keeping
Influential Neighbour Sets	R*-tree VoR-tree	Moving kNN query	safe guarding objects Influential Neighbour Sets
Data Frame Based Spatiotemporal Indexing	Spatiotemporal Index	Moving Objects Range query	spatiotemporal trajectory retrieval algorithm Geohash based spatial index
D-Grid	dual space grid index	Moving Objects Range and kNN query	indexes moving objects using grid structures in both location and velocity spaces

In this paper the hybrid indexing algorithm based on grid index is proposed which is used to answer the present Geo-Textual moving queries executed on Geo-Textual moving objects whose velocity vector is not known well in advance [20]. The hybrid indexing algorithm based on grid index and inverted list, focuses on reducing the nearest neighbours query processing time delay by pruning the search space.

III. METHODOLOGY

The most of the research done in past for moving queries on moving objects is based on known motion pattern of the objects and queries. The representative algorithms which do not assume any moving pattern are YPK-CNN, SEA-CNN and CPM [18]–[20]. These algorithms use grid index to index dynamic data, to capture the updated location of the objects and to maintain the index. In the proposed methodology, these 3 indexing techniques are converted to geo-textual hybrid index and their performance is studied against non-hybrid implementation to study the improvement due to proposed techniques.

Geospatial query optimization for moving objects needs to focus on the following issues:

- the need to update the index as objects move
- the need to re-evaluate affected queries when any object moves
- the need to re-evaluate moved queries and
- achieving very short execution times for large numbers of moving objects and queries

$$T_{rtpc} = T_{upd\_ind} + T_{recompute\_query} \quad (1)$$

In equation 1, T_{rtpc} is the running time per processing cycle. It consists of index maintenance time (T_{upd_ind}) and the query re-evaluation time ($T_{recompute\_query}$) for all affected and moved queries during the processing cycle. The objective of the research is to minimize $T_{rtpc.}$

In this research the moving objects and query points are handled where the objects' movements are non-predictable, i.e., their velocity vectors and motion patterns are unknown.

The query points and the data objects move frequently and arbitrarily and multiple queries have to be processed simultaneously. Constant reporting of its results is essential and hence geo-textual hybrid indexing techniques are used to minimize the index maintenance and query execution time. In tree-based indexes, the continuous moving of objects cause the nodes of the index structure frequently be split or merged, which certainly leads to high maintenance cost. Also in main memory R-tree very expensive to maintain dynamically. Hence this research focuses on Grid-Inverted List hybrid index to optimize the maintenance and query execution cost. In figure 1, the steps for index creation/maintenance are query execution are shown. The boxes marked in blue are the area of focus in this research.

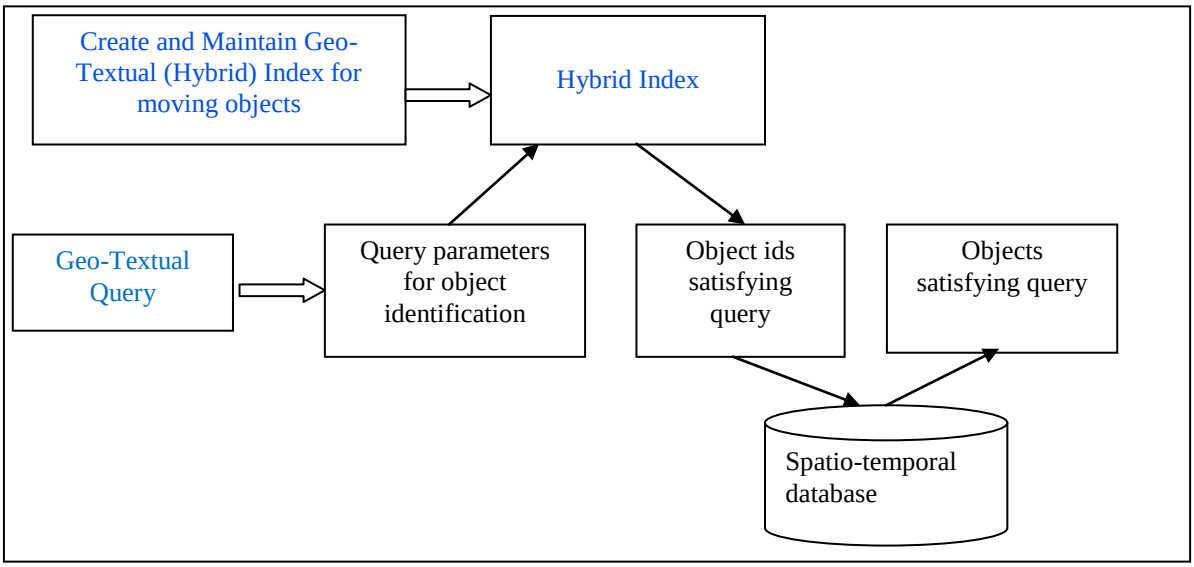

Figure 1 index creation/maintenance and query execution steps

Following 4 proposed algorithms are developed, implemented and studied to evaluate their performance:

1. YPK-CNN search technique – Grid-Inverted List Hybrid Index structure is developed to implement YPK-CNN techniques to answer Geo-Textual kNN queries
2. SEA-CNN search technique – Grid-Inverted List Hybrid Index structure is developed to implement SEA-CNN techniques to answer Geo-Textual kNN queries
3. CPM search technique – Grid-Inverted List Hybrid Index structure is developed to implement CPM techniques to answer Geo-Textual kNN queries
4. CPM search technique - implemented as non-Hybrid Index Structure to compare hybrid and non-hybrid indexes

In this paper the CPM technique is explained as it is proved to be efficient among the 4 techniques.

IV. GRID-INVERTED LIST HYBRID INDEX

The grid indices divide space into a predefined number of equal-sized square or rectangular cells which is created using a 2-dimensional array. Each cell of the array represents a region of a space [18]. Grid – Inverted list index combines a grid index with the inverted list to organize Geo-Textual objects. The grid index and the text index are combined tightly in way that every grid cell has the inverted list associated with it for all the objects in this region. Each cell C of the grid is associated with the list of objects within its extent and the inverted list of the keywords of the objects belonging to it. Each inverted list has vocabulary of all distinct words appearing in description of the objects belonging to the cell and a posting list for each word. Figure 2 illustrates hybrid index structure in which each cell also has influence list of the queries. The entry of a query in influence list of the cell is meant for book keeping, which notifies that, this cell is in the influence region of the query and any changes to it will affect the query and hence it needs to be updated.

Tight combination of grid file with Inverted list helps in pruning the search space efficiently as it does not have to check all the objects for answering query but only the objects which satisfy text criteria. Data space is mapped to grid cells by normalizing the latitude and longitude of objects to x and y coordinates of the grid. Each Geo-Textual object has latitude and longitude describing the point location and a text description of the object. Latitude and longitude is mapped to the grid cell depending upon the size (m x n) of the grid structure. Each object is added to the object list of the grid cell it is mapped to and inverted list of the corresponding cell is updated for the keywords of the object.

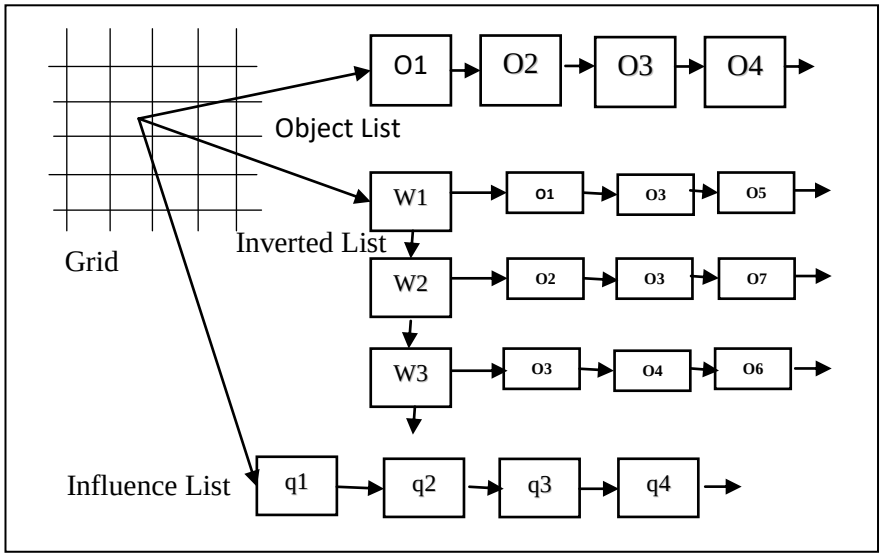

Figure 2 Grid-Inverted List Hybrid Index Structure

V. CONCEPTUAL PARTITIONING

Conceptual partitioning as described by Mouratidis et al. is used to find k nearest neighbours (kNN) of the query point efficiently [20]. Query object has latitude and longitude describing the point location and a set of keywords. Latitude and longitude of a query point q is mapped to the gird cell Cq. The mindist(C, q) is the minimum distance between any object p ∈ C and the query point q. The bestkNN denotes the list of k best nearest neighbours of q found so far maintained in the order of mindist. The bestdist is the distance between the kth NN and q.

The naive way to process kNN for q has to sort all cells C ∈ G in mindist(C,q) order and visit them in ascending order. It is optimal with respect to the number of processed cells but is expensive because it must find mindist for all cells and sort them. Conceptual partitioning and book keeping avoids unnecessary computations by partitioning the grid space and doing computations only if required.

kNN algorithm visits cells and rectangles in ascending order of mindist(C, q) by processing the minimal set of cells. kNN algorithm maintains the priority queue (PQ) with the nodes in the increasing order of mindist as the key. The node either has the boundary coordinates for a cell or for a rectangle.

Algorithm 1 shows the algorithm to compute kNN. Initially the cell Cq is inserted in the PQ with mindist(Cq, q) = 0 (line 5). Then all level 1 rectangles are inserted in PQ with mindist(DIR1, q) (line 6). Then it starts removing the elements iteratively from the PQ (line 7, 8). If the removed element is a cell and there are objects inside the cell, it checks inverted list for the cell (line 9, 10). If inverted list satisfies the query keywords, then for objects satisfying query criteria in the object list of the cell, dist(p, q) is calculated and bestkNN is updated for this object (line 11). Elements removed from PQ are inserted into the visit list of the query (line 12). Query is added to the influence list of every visited cell to mark the influence region of the query (line 13). If the removed element is a rectangle, it inserts each cell of the rectangle in the PQ with its mindist(C, q) as key (line 15) and inserts the next level rectangle in same direction with its mindist (line 16). Algorithm terminates when the PQ is empty or bestkNN has k objects and the next entry in PQ has key greater than bestdist (line 17).

Conceptual partitioning gives correct result because (i) the PQ has the key as mindist which is arranged in the ascending order. (ii) Every rectangle that is inserted in the PQ has the mindist value which is greater than the rectangles inserted till now with no rectangle missed out. (iii) these rectangles in the PQ act as bounding boxes. While computing kNN, if it is found that mindist(C,q) > bestdist, C can be safely pruned because the objects contained in C will be farther than all the current kNN's of q.

Algorithm 1 kNN Search Algorithm

```
kNN_search(G, q, k)
Input:
G-Grid Index
q- query point (latitude, longitude, keywords)
k – number of nearest neighbours
Output:
Set of k nearest neighbors- bestkNN
Algorithm:
1.  bestdist = ∞
2.  bestkNN = NULL
3.  PQ = NULL
4.  visitlist = NULL
5.  Insert query cell <Cq, 0> into PQ
6.  For each direction of the query cell insert rectangle
      <DIR1, mindist(DIR1, q)> into PQ
7.  While (!empyt(PQ) and key of PQ <= bestdist)
8.      remove element from PQ
9.      if it is cell element
10.         if inverted list of the cell satisfies
            query keywords
11.             update bestkNN and bestdist if necessary
                for the objects in cell
12.             insert removed element in the visitlist
13.             insert q in the influence list of the cell
14.     else
15.         insert each cell of the rectangle into PQ
16.         insert next level rectangle in the same
            direction into PQ
17. return bestkNN
```

Figure 3 depicts the example of how the cells are typically visited with conceptual partitioning. Cell numbered 1 is the query cell, which is visited first. Then the surrounding cells are visited in the increasing order of minimum distance of the cell from the query point. With this nearer cell (hence objects) are visited before visiting the farther cells (hence objects). Inverted list in each cell further restricts the search space by only visiting cells and objects which satisfy the query criteria.

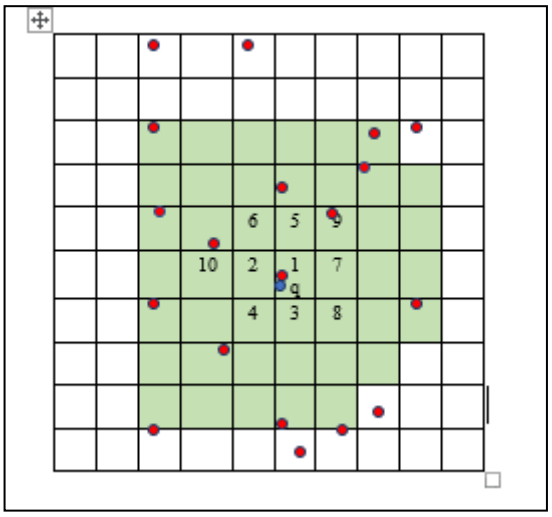

Figure 3 Conceptual Partitioning and Visiting order

VI. HANDLING MOVING OBJECTS AND QUERIES

Moving objects and queries are handled in every update cycle. In each update cycle first the moved objects during the time interval are updated in the index structure by removing them from earlier cell and adding to the new moved cell. Along with this inverted list and influence lists are also updated. To handle queries book keeping is done in every update cycle. The running queries are stored along with their current results in a query table. This query table is updated while updating index structures for moving objects and answering moving queries. Visit list in the query table is used to update the query results when there are any updates in the influence region of the query due to movements of the objects during the update cycle. Figure 4 shows the structure of the query table used for book keeping. The Influence list is maintained in cells of a grid structure to keep track of the influence region of queries. Any movement in influence region of the query will affect the query and hence we need to only focus on the queries which have affected during current update cycle for updating the query answers.

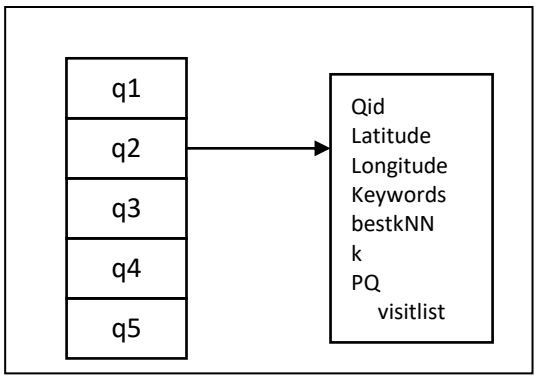

Figure 4 Structure of Query table

Algorithm 2 shows the algorithm to update kNN when the query is affected due to movements of objects in the influence region of the query. Visit list is traversed to visit each cell, the inverted list of the cell is checked for query keywords (line 3, 4). If inverted list satisfies the query keywords, then for objects satisfying query criteria in the object list of the cell, dist(p, q) is calculated and bestkNN is updated for this object (line 5). If bestkNN has less than k objects priority queue of the query is processed to further find the NN objects (line 6, 7).

Algorithm 2 kNN update algorithm

```
Update_search(q, visitlist, k)
Input:
Visitlist – visit list of the query
q- query point (latitude, longitude, keywords)
k – number of nearest neighbours
Output:
Set of k nearest neighbors- bestkNN
Algorithm:
1.  bestdist = ∞
2.  bestkNN = NULL
3.  For each cell in the visitlist
4.      if inverted list of the cell satisfies
        query keywords
5.          update bestkNN and bestdist if necessary
            for the objects in cell
6.  if bestkNN has less than k objects
7.      process PQ to find more NN objects
8.  return bestkNN
```

VII. EXPERIMENTAL WORK

Sample data sets having objects with their location (Latitude and Longitude) and attribute data are used for evaluating Grid-Inverted list hybrid index and Geo-Textual kNN query. Parameters used for evaluating algorithms are number of objects (N), number of simultaneous queries(n), number of nearest neighbour (k) and grid size (δ). Experiments are carried out with varying number of nearest neighbours and with 5 different grid sizes [5x5, 10x10, 15x15, 20x20, 25x25]. Work is evaluated by moving or terminating some objects and queries in each update cycle and also by introducing new objects and queries.

The results are compared by implementing 2 other variations in grid index used for moving objects (YPK-CNN, SEA-CNN) by converting them to Grid - Inverted List hybrid index structures and also with one non-hybrid Grid index structure [18][19].

VIII. RESULTS AND DISCUSSION

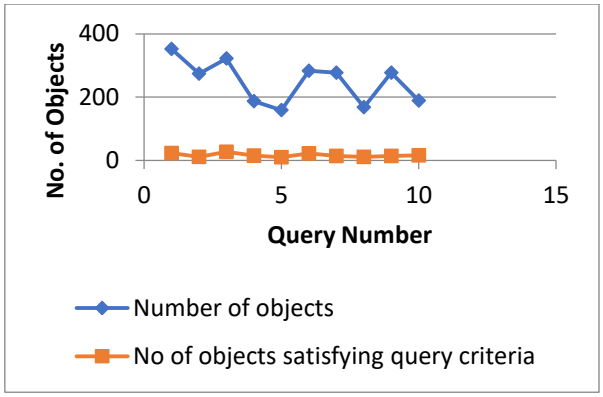

Figure 5 Number of Objects and number of objects satisfying query criteria

Figure 5 shows the actual number of objects evaluated and the number of objects satisfying the query criteria. Numbers of actual hits (typically maximum 9% as in figure 5) are reduced and hence false hits are avoided by using inverted index in hybrid index structure thus pruning the search space.

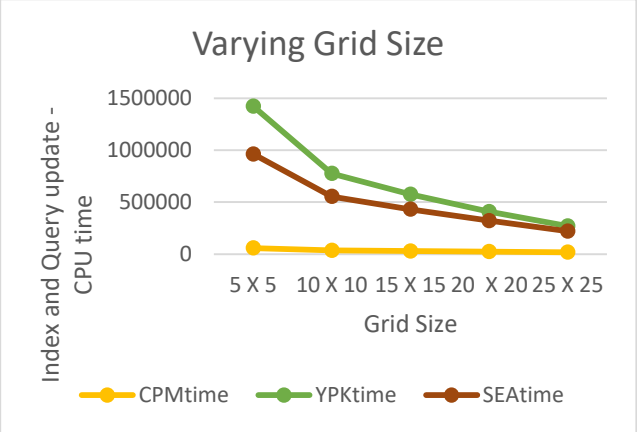

Figure 6 Grid Size vs Performance

Figure 6 shows the effect of changing the grid size. when the number of cells is less, the length of object list and inverted list is more and hence it must visit more number of nodes. As the number of cells are increased the objects are distributed among more number of cells and hence the time taken is less.

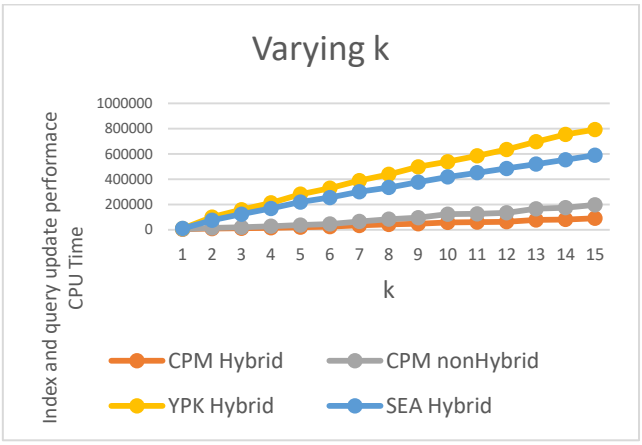

Figure 7 k vs performance

Figure 7 show as the number k is increased the time taken is more. This is due to the fact that the more number of cells are visited to find more number of nearest neighbours.

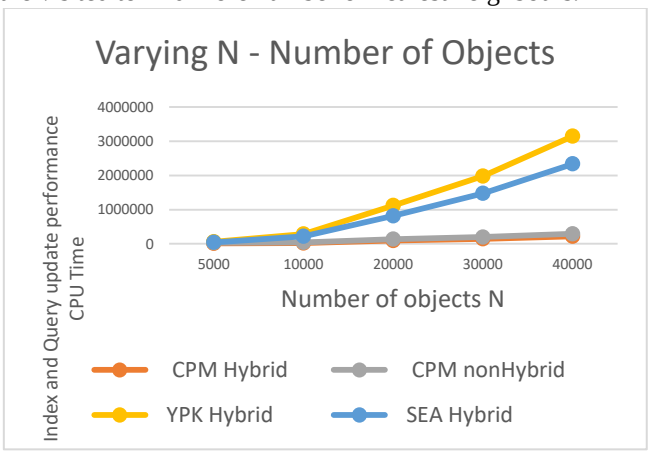

Figure 8 N vs Performance

Figure 8 displays the effect increasing number of objects, N on index and query update performance. As can be seen from the graph, as the number of objects increase, the time taken to update index and answer query is more.

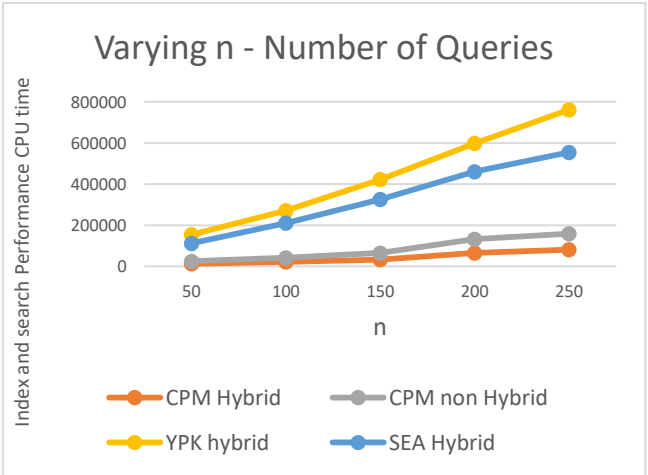

Figure 9 n vs performance

As shown in figures 9, the query result maintenance time in a particular update cycle grows linearly with number of queries, n, as more number of queries are processing during the particular update cycle.

$$T_{build} = T_{index} + T_{query} \quad (2)$$

The run time T_{build} (equation 2) of building, computing query answers consists of T_{index} and T_{query}. T_{index} is the time required to build the hybrid Index and it depends upon the number of objects in dataset. T_{query} is the time of computing the k-NN using conceptual partitioning. It consists of Heap creation and handling, Number of cells visited and the number of objects evaluated.

$$T_{update} = T_{upd\_ind} + T_{recompute\_query} \quad (3)$$

The run time T_{update} (equation 3) is the time required to update index and to re-evaluate affected queries in every update cycle. T_{upd_index} is the time required to update hybrid-index for moving objects by using book keeping information. It depends upon the number of updates and type of updates. $T_{recompute\_query}$ is the time required for re-computing query by using book keeping information and hybrid index. The time required to re-compute query is proportional to the number of cells visited and the number of objects evaluated. Figure 7 highlights the time taken to maintain index and re-compute query for various values of k, i.e. number of nearest neighbours for 4 different methods. It is observed that the performance gain seen by hybrid index and CPM is 30 to 40 percent as compared to the other methods. Figure 6 also shows the improvement in performance by increasing the grid size. maintaining index and computing query answers.

IX. CONCLUSION

The paper proposes the solution for creating and maintaining hybrid index based on uniform grid and inverted list for moving objects. Uniform grid makes easy to calculate the cell of given object with its latitude and longitude in constant time. Grid maintenance is easier for the moved objects in each update cycle. In case the object is moved to different cells, it is just removing the object from old cell and adding it to the new cell.

The method is proposed to find k nearest neighbours of static and moving queries using hybrid index and conceptual partitioning of the grid. With the hybrid index having grid and inverted lists tightly combined, the search space narrows down as compared to each index applied separately. It reduces the number of false positive hits to database records. Conceptual partitioning restricts the space to be searched around the query point and helps in retrieving objects in efficient manner in kNN processing. kNN monitoring using book keeping (Query Table, Influence List) information for moving queries and objects further reduces the search time.

The extensive experiments have demonstrated reducing the update and search cost for uniform grid-inverted list hybrid index. Hybrid index and conceptual partitioning together improve the query performance by 30 to 40%.

Future work can further improve the performance by implementing the multilevel grid index for skewed data. Search algorithm in the paper uses Euclidean space, which can be further studied for road networks. kNN search query algorithm is presented in the paper, whereas other type of Geo-Textual queries on hybrid index is in the further research scope. Keyword queries can further be studied for natural language processing (NLP) queries, fuzzy queries and other variations.

REFERENCES

1. Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma, "Hybrid index structures for location-based web search," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, 2005, pp. 155–162.
2. I. De Felipe, V. Hristidis, and N. Rishe, "Keyword search on spatial databases," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, 2008, pp. 656–665.
3. Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, and D. L. Lee, "IR-Tree: An Efficient Index for Geographic Document Search," p. 15.
4. G. Cong, C. S. Jensen, and D. Wu, "Efficient retrieval of the top-k most relevant spatial web objects," *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 337–348, Aug. 2009, doi: 10.14778/1687627.1687666.
5. M. Christoforaki, J. He, C. Dimopoulos, A. Markowetz, and T. Suel, "Text vs. space: efficient geo-search query processing," in *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, Glasgow, Scotland, UK, 2011, p. 423, doi: 10.1145/2063576.2063641.
6. S. N. Aung and M. Mint Sein, "Hybrid Geo-Textual Index Structure for Spatial Range Keyword Search," *Comput. Sci. Eng. Int. J.*, vol. 4, no. 6, pp. 21–28, Dec. 2014, doi: 10.5121/cseij.2014.4603.
7. S. Vaid, C. B. Jones, H. Joho, and M. Sanderson, "Spatio-textual Indexing for Geographical Search on the Web," in *Advances in Spatial and Temporal Databases*, vol. 3633, C. Bauzer Medeiros, M. J. Egenhofer, and E. Bertino, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 218–235.
8. A. Khodaei, C. Shahabi, and C. Li, "Hybrid Indexing and Seamless Ranking of Spatial and Textual Features of Web Documents," in *Database and Expert Systems Applications*, vol. 6261, P. G. Bringas, A. Hameurlain, and G. Quirchmayr, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 450–466.
9. X. Cao, G. Cong, C. S. Jensen, and M. L. Yiu, "Retrieving regions of interest for user exploration," *Proc. VLDB Endow.*, vol. 7, no. 9, pp. 733–744, May 2014, doi: 10.14778/2732939.2732946.
10. L. Chen, G. Cong, and X. Cao, "An efficient query indexing mechanism for filtering geo-textual data," in *Proceedings of the 2013 international conference on Management of data - SIGMOD '13*, New York, New York, USA, 2013, p. 749, doi: 10.1145/2463676.2465328.
11. D. Zhang, K.-L. Tan, and A. K. H. Tung, "Scalable top-k spatial keyword search," in *Proceedings of the 16th International Conference on Extending Database Technology - EDBT '13*, Genoa, Italy, 2013, p. 359, doi: 10.1145/2452376.2452419.
12. C. Zhang, Y. Zhang, W. Zhang, and X. Lin, "Inverted Linear Quadtree: Efficient Top K Spatial Keyword Search," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1706–1721, Jul. 2016, doi: 10.1109/TKDE.2016.2530060.
13. J. F. G. Li, L. Z. S. Chen, and J. Hu, "SEAL: Spatio-Textual Similarity Search," p. 12.
14. D. V. Kalashnikov, S. Prabhakar, and S. E. Hambrusch, "Main Memory Evaluation of Monitoring Queries Over Moving Objects," *Distrib. Parallel Databases*, vol. 15, no. 2, pp. 117–135, Mar. 2004, doi: 10.1023/B:DAPD.0000013068.25976.88.
15. Ying Cai, K. A. Hua, and Guohong Cao, "Processing range-monitoring queries on heterogeneous mobile objects," 2004, pp. 27–38, doi: 10.1109/MDM.2004.1263040.
16. 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 Trans. Comput.*, vol. 51, no. 10, pp. 1124–1140, Oct. 2002, doi: 10.1109/TC.2002.1039840.
17. 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.
18. Xiaohui Yu, K. Q. Pu, and N. Koudas, "Monitoring k-Nearest Neighbor Queries over Moving Objects," 2005, pp. 631–642, doi: 10.1109/ICDE.2005.92.
19. 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, doi: 10.1109/ICDE.2005.128.
20. K. Mouratidis, D. Papadias, and M. Hadjieleftheriou, "Conceptual partitioning: an efficient method for continuous nearest neighbor monitoring," 2005, p. 634, doi: 10.1145/1066157.1066230.
21. Simonas ˇSaltenisy Christian S. Jenseny Scott T. Leuteneggerz Mario A. Lopezz, "Indexing the Positions of Continuously Moving Objects." ACM.
22. R. Benetis, C. S. Jensen, G. Karciauskas, and S. Saltenis, "Nearest neighbor and reverse nearest neighbor queries for moving objects," 2002, pp. 44–53, doi: 10.1109/IDEAS.2002.1029655.
23. C. Shahabi, "Time Parameterized Queries in Spatio Temporal Databases," p. 31.
24. Y. Tao, D. Papadias, and J. Sun, "The TPR*-Tree: An Optimized Spatio-Temporal Access Method for Predictive Queries," p. 12.
25. K. Raptopoulou, A. N. Papadopoulos, and Y. Manolopoulos, "Fast Nearest-Neighbor Query Processing in Moving-Object Databases," p. 25.
26. G. S. Iwerks, H. Samet, and K. P. Smith, "Maintenance of *K* -nn and spatial join queries on continuously moving points," *ACM Trans. Database Syst.*, vol. 31, no. 2, pp. 485–536, Jun. 2006, doi: 10.1145/1138394.1138396.
27. An Chunming and Li Zongsen, "Studies on kNN query of moving objects for location management in spatial database," 2013, pp. 2428–2432, doi: 10.1109/MEC.2013.6885443.
28. Y.-K. Huang, Z.-H. He, C. Lee, and W.-H. Kuo, "Continuous Possible K-Nearest Skyline Query in Euclidean Spaces," 2013, pp. 174–181, doi: 10.1109/ICPADS.2013.35.
29. Y. Park *et al.*, "A New Spatial Index Structure for Efficient Query Processing in Location Based Services," 2010, pp. 434–441, doi: 10.1109/SUTC.2010.64.
30. C. Li, Y. Gu, J. Qi, G. Yu, R. Zhang, and Q. Deng, "INSQ: An influential neighbor set based moving kNN query processing system," 2016, pp. 1338–1341, doi: 10.1109/ICDE.2016.7498339.
31. M. F. Mokbel, X. Xiong, and W. G. Aref, "SINA: scalable incremental processing of continuous queries in spatio-temporal databases," 2004, p. 623, doi: 10.1145/1007568.1007638.
32. 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.
33. C. Li, Y. Gu, J. Qi, G. Yu, R. Zhang, and W. Yi, "Processing moving *k* NN queries using influential neighbor sets," *Proc. VLDB Endow.*, vol. 8, no. 2, pp. 113–124, Oct. 2014, doi: 10.14778/2735471.2735473.
34. 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, 2016, pp. 2592–2597, doi: 10.1109/WCICA.2016.7578643.
35. 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, 2016, pp. 252–261, doi: 10.1109/MDM.2016.46.
36. 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: 10.1109/TKDE.2017.2742956.

AUTHORS PROFILE

Sulbha Kishor Powar is a PhD Research Scholar under the supervision of Dr. Ganesh Magar at Post Graduate Department of Computer Science, S.N.D.T. Women's University, Santacruz (West), Mumbai- 400 049, Maharashtra, India. Her research area is Geo-Textual query processing. Her fields of interest are Data structures, Databases, Programming languages, Geographic Information

Systems, Data Mining. She has 26 years of work experience in development of various Information Systems at university level. She also has experience of teaching and guiding students at Masters' level and conducting training programs. She is M.Sc. (Electronics) from Mumbai University. She has done professional courses, Post Graduate Diploma in Software Technology (PGDST) and Certificate Course in Software Technology (CCST) from National Centre for Software Technology (NCST – Now CDAC).

Dr. Ganesh M. Magar is an Associate Professor at Post Graduate Department of Computer Science, S.N.D.T. Women's University, Santacruz (West), Mumbai- 400 049, Maharashtra, India. He has done Ph.D. in Computer Science. He has 2 years of experience in IT industry and 19 years of experience in teaching. His fields of interest are Geographical Information System (GIS), Web GIS, Mobile GIS, Geo Spatial BIG DATA Analysis, Image Processing and SAR Imaging, Human Computer Interaction, Visual Data Mining. He is involved in syllabus design committee of the University for Under Graduate and Post Graduate level courses. He is member of various professional bodies like ISCA, CSI, ISTE, ISRS, IEEE.