International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:8, No:1, 2014

# A Robust Method for Finding Nearest-Neighbor using Hexagon Cells

Ahmad Attiq Al-Ogaibi, Ahmad Sharieh, Moh'd Belal Al-Zoubi, R. Bremananth

*Abstract*—In pattern clustering, nearest neighborhood point computation is a challenging issue for many applications in the area of research such as Remote Sensing, Computer Vision, Pattern Recognition and Statistical Imaging. Nearest neighborhood computation is an essential computation for providing sufficient classification among the volume of pixels (voxels) in order to localize the active-region-of-interests (AROI). Furthermore, it is needed to compute spatial metric relationships of diverse area of imaging based on the applications of pattern recognition. In this paper, we propose a new methodology for finding the nearest neighbor point, depending on making a virtually grid of a hexagon cells, then locate every point beneath them. An algorithm is suggested for minimizing the computation and increasing the turnaround time of the process. The nearest neighbor query points $\Phi$ are fetched by seeking fashion of hexagon holistic. Seeking will be repeated until an AROI $\Phi$ is to be expected. If any point $\Upsilon$ is located then searching starts in the nearest hexagons in a circular way. The First hexagon is considered be level 0 ($L_0$) and the surrounded hexagons is level 1 ($L_1$). If $\Upsilon$ is located in $L_1$, then search starts in the next level ($L_2$) to ensure that $\Upsilon$ is the nearest neighbor for $\Phi$. Based on the result and experimental results, we found that the proposed method has an advantage over the traditional methods in terms of minimizing the time complexity required for searching the neighbors, in turn, efficiency of classification will be improved sufficiently.

*Keywords*—Hexagon cells, $k$-nearest neighbors, Nearest Neighbor, Pattern recognition, Query pattern, Virtually grid.

## I. Introduction

FINDING nearest neighbor points among volume-of-pixels (voxels) is pertinent computationally challenge issue in many statistical classification applications. In the searching of regions-of-interests, segmentation decays an image into ingredient regions, acquaintance of boundaries, or objects to be seeking. The level to which decomposition is performed based on the problem being solved in applications such as automation inspection using images and interest lies in analyzing objects' outlines. These applications currently suffers due to deplorable computation turnaround time of traditional nearest neighbor method such as $k$-nearest neighbors ($k$-NN). In $k$-NN, $k$ depends on the number of nearest neighbors utilized in the classification while seeking for the objects. Especially, in remote sensing, user control over imaging is inadequately restricted due to the choice of image sensors, for these applications, seeking of boundaries mainly depends on segmentation. However, efficacy of classification mainly depends on accuracy of segmentation accuracy.

Ahmad Attiq and Mohammad Al-Zoabi are with the University of Jordan. Ahmad Sharieh, and R Bremananth are currently with the Information Systems and Technology Department, Sur University College, Sur, P. O.Box 440, PC. 411, Sultanate of Oman. Corresponding authors can be reached at ahmadsharieh@suc.edu.om, bremresearch@gmail.com or bremananth@suc.edu.om (Let you know more, see http://www.suc.edu.om).

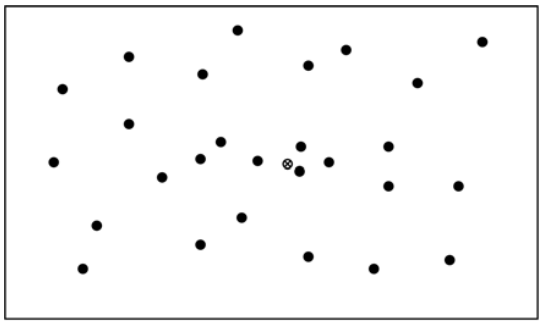

Fig. 1. $\aleph$ is the closet to query point $\Phi$ in n points 2-D space. In that, $\Phi$ is represented by the small circle with cross inside it, while data points are represented by the small filled circles.

Optimize the accuracy of $k$-nearest neighbor classification based on the weighted distance was presented in [8]. A selective incremental approach for transductive Nearest Neighbor Classification was proposed by Viswanath et al. [10]. In this paper, we provide a solution to classify the patterns by using hexagonal searching of neighbors.

We can treat a set of $\aleph$ number of n points in two-dimensional (2-D) spaces, in that, a point $\aleph$ is a nearest point to a query point, $\Phi$. This is represented by a cross encircle enclosed by a circle in Fig. 1. [4]

The proposed method starts in preprocessing phase by making a virtually grid of a hexagon cells, then locate every point beneath them. In order to find the nearest neighbor to a query point $\Phi$, a search on the same hexagon is performed until $\Phi$ is expected to be found. However, if any point $\Upsilon$ is found while searching, then circular way of seeking is required for the nearest hexagons. An initial hexagon is called level 0 $L_0$ and the surrounded hexagons is referred as level $L_1$. If any point $\Upsilon$ is not found in $L_1$ then search is enforced to the next level $L_2$ in order to ensure that $\Upsilon$ is the nearest neighbor for $\Phi$, as shown in Fig. 2.

The mathematical model of the proposed method can be summarized as follows: A given set $\aleph$ has n number of points in 2-D space. It is represented by small filled circles as shown in Fig. 1, and a query point $\Phi$ which is denoted by small circle with cross inside. The problem is to find the nearest $\Phi$, from the points $\aleph$ (i = 1, 2, 3... n) by starting from initial point $\omega$.

In preprocessing phase, a virtually grid of a hexagon cells over $\omega$ is drown as shown in Fig. 3a. The located points are underneath on every hexagon and store them in a linked list while implementation. The list shows every hexagon with the points beneath on it and every point which hexagon belongs

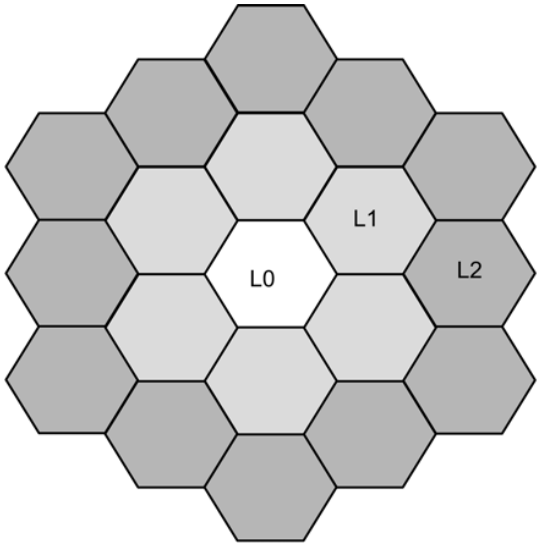


Fig. 2. Representation of hexagonal space of points to be searched for the given query point $\Phi$.

to. After that, locate $\Phi$ in a hexagon at level $L_0$, then searching is continued inside boundary of the hexagon. If any point $\Upsilon$ is found, then searching in surrounded hexagons $L_1$ is carried out for nearest point to $\Phi$ rather than found in the same hexagon. This is shown in Fig. 3b. If $\Phi$ is considered as matching of query point, then the initial point found in one of the surrounded hexagons, for example, $L_1$, as shown in Fig. 3c. A search in the remaining hexagons in the same level is repeated until the next level $L_2$ is made. Our contribution is to search in two levels of hexagons in order to find the nearest point for the query point $\Phi$ in the set $\aleph$ of data. This is a easier way than drawing circles and closer to the circle shape than square, and it is easier way to search than exhaustive method and speed up the process.

The remainder of this paper is organized as follows. Section II describes state-of-art techniques related with neighbor classification and nearest neighbor seeking problems. Proposed methodology is given in Section III. Section IV depicts the result and analysis based on the benchmark of data set in term of time and space complexities. Concluding remarks and future enhancement are suggested in Section V.

## II. Literature Review

There are many state-of-art methods which are related with neighborhood searching and classification problems. Based on the review of previous literature, we can faction the neighborhood operation issues by two extensive categories such as seeking of query points related to its neighbors and classification based on the previously classified trained set.

In [8], a method was proposed for nearest neighbor classification rules based on weighted distance. These weights fixation specific for each unambiguous class, feature set for the classification and characteristics of the archetype. The proposed learning algorithms are resulted by minimizing the Leaving-One-Out-Cross Validation-classification error (LOOCVCE) of the given training set.

A transductive method was proposed for nearest neighbor classification through non-spectral or spectral graph minimum cuts for the rouge patterns [10]. Solution to these issues is achieved based on the LOOCVCE reverence to base classifiers such as support vectors machine (SVM), $k$-NN and others. A zero LOOCVCE can be achieved through a clustering method, however it has a less classification efficacy for diverse noise patterns which are involved in the training and testing processes. Lawrence et al. proposed a learning framework for fast retrieval of nearest neighbor points based $k$-dimensional tree space-partitioning data structure [4]. In [6], Nearest neighbor imputation of species-level was proposed for analyzing ground and remotely sensed data measured comprehensively across the same forested landscape.

In [12], k-nearest neighbor text classification algorithm based on fuzzy integral was proposed. The proposed method avoided independence demand of Dempster-shafter theory (D-S theory) and enhanced the performance of text classification.

David et al. [7] described a method to find the closest match within a database of other such items, which was a task performed in numerous domains. Image matching, data mining, and electroencephalogram data analysis are a few varied examples. They have used the extension of the concept of Euclidean distance in 2D and 3D space to higher dimensional space provides an effective comparison of items in these sorts of domains.

In [11], a method was suggested to find nearest neighbors using $k$-nearest neighbors algorithm ($k$-NN). It adopted in the development of a general methodology, neighborhood counting, for devising similarity functions. In order to measure the similarity between two data points, all neighborhoods are considered to cover both data points. Neighborhood can be defined for different types of data for multivariate data and derived a formula for such similarity, called neighborhood counting measure (NCM). The NCM was tested experimentally in the framework of $k$-NN. In the experiments, the NCM consistently outperforms a mixture of Euclidean and Hamming distances (HEOM). The NCM has a computational complexity in the same order as the standard Euclidean distance function. The NCM was proven the sound for multivariate data experimentally.

In [5], a method proposed for computing the given set S of points in a metric space with distance function D. In order to solve the nearest-neighbor searching problem, a data structure was built for S. So that for an input query point q, the point s to S that minimizes D(s, q) can be found rapidly. Several measures of dimension can be estimated using nearest-neighbor searching, while others can be used to estimate the cost of the searching with low-dimensional spaces.

In [9], Sun, et al. proposed a way to incorporate the nearest neighbor search problem based on analysis of algorithms course. The problem of searching the elements of a set that are close to a given query element under some similarity criterion has a vast number of applications in many branches such

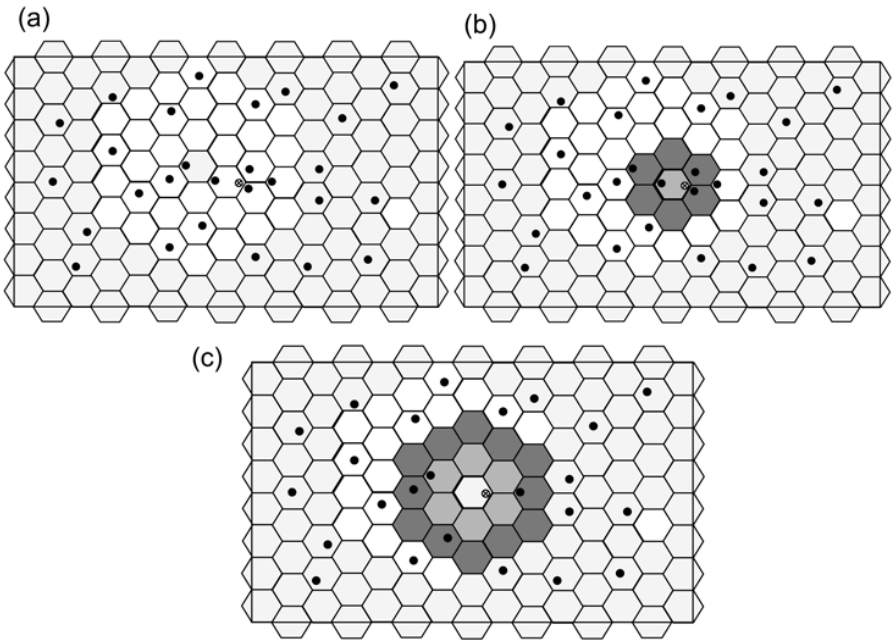


Fig. 3. (a) Drawing a virtually hexagon cells on ℵ set. (b) Searching in the same Hexagon, if found one we search in surrounded hexagons for nearest point than found in the same hexagon. (c). If we found the first point in $L_1$ we Search in the remain hexagons in the same level, and all the hexagons in the next level.

as pattern recognition to textual and multimedia information retrieval. Many solutions have been proposed in different areas, in many cases without cross-knowledge. Because of this, the same ideas have been preconceived several times, and very different presentations have been given for the same approaches. They presented some basic results that explain the intrinsic difficulty of the search problem. Sunil Arya et al. suggested an optimal algorithm for approximating nearest neighbor searching in fixed dimension in [2].

In [1], they proposed the idea of using the hexagonal search space.

From previous review and from our thesis [1], it can be noticed that all methods used to find the nearest neighbors are a traditional methods depending on either probability principals or analytical methods, while the proposed method, in this research will be forming a new paradigm, depending on drawings, geometry and computationally a simple calculations. Based on the result comparisons, we found that the proposed method will make searching query point in a robust way and around 20-30% faster than existing methods.

## III. Theoretical background and Problem formulation

The problem of finding the neighborhood center is involved on image data arbitrary dimensionality. It doesn't represent in terms of color or intensity of pixels instead, it is a structure of related volume-of-pixels (voxels). Region of voxels dependents on which search is made by the operation. That is, voxels is either a fixed size square or a cube depending on the dimensionality of the image data. So that, neighborhood searching is needed of arbitrary dimension of data structure to accomplish the center point searching among the neighbor points. In our paper, we have utilized $k$-dimensional tree (kd-trees) data structure for performing exact and approximate nearest neighbor searching. Bentley J. L. introduced the kd-trees as a generalization of the binary search tree in higher dimensions [3]. Each node of the tree is implicitly associated with a d-dimensional rectangle that is called a cell. The root node is associated with the bounding rectangle, which encloses all of the data points. Each node is also implicitly associated with the subset of data points that lie within this rectangle. If the number of points associated with a node falls below a given threshold, called the bucket size, then this node is a leaf, and these points are stored with the leaf.

Based on the fundamental of kd-trees, we contribute the method of hexagonal data structure in order to decrease the turn around time of the each query point over the d-dimensional space. In our experiments, we utilized a bucket size of one. Our splitting algorithm is based on the hexagonal points as stated in Fig. 3. Queries are answered by a recursive algorithm. In the basis case, when the algorithm arrives at a leaf of the tree, it computes the distance from the query point to each of the data points associated with this node. The smallest such distance is saved. When arriving at an internal node, it first determines the side of the associated hexagonal on which the query point lies.

On returning from the seeking, it is to determine that whether the cell is associated with the other siblings which is closer to the query point or not. If it is closer to the query point, then the seeking is repeated. If existence of query point, then this sibling and leafs are also visited recursively. When the search returns from the root, the closest point is

returned. An important observation is that for each query point, every leaf whose distance from the query point is less than the nearest neighbor will be visited by the algorithm. In order to generalization, seeking algorithm is approximated to the nearest neighbor queries. Let $\Psi$ denote the allowed error bound. In the processing of an internal node, the further child is visited only if its distance from the query point is less than the distance to the closest point so far, divided by $1+\Psi$.

### A. Construction of Hexagons and Algorithm

In order to construct hexagonal formulation, we model the neighbors of pixels as illustrated in Fig. 4. It is started by constructing a hexagonal ABCDEF and specifying the center of the hexagon.

**Step 1.** The initial step starts by constructing the hexagon on 2-D space, as shown in Fig. 4

**Step 2.** The second step is the preprocessing i.e. going to the center of the hexagon, and the heads of this hexagon using equations: (1)-(6) .

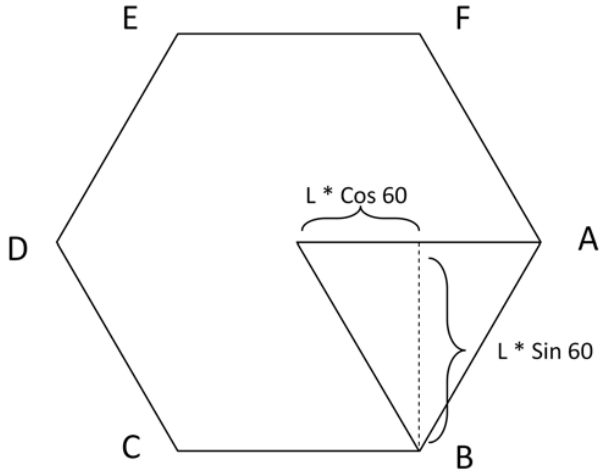


Fig. 4. Representation of constructing hexagon neighborhood Points ABCDEF by searching the pixel space in the image.

$$A = (x+L, y), \quad (1)$$
$$B = (x+L\cos(\frac{\pi}{3}), y+L\sin(\frac{\pi}{3})), \quad (2)$$
$$C = (x-L\cos(\frac{\pi}{3}), y+L\sin(\frac{\pi}{3})), \quad (3)$$
$$D = (x-L, y), \quad (4)$$
$$E = (x-L\cos(\frac{\pi}{3}), y-L\sin(\frac{\pi}{3})), \quad (5)$$
$$F = (x+L\cos(\frac{\pi}{3}), y-L\sin(\frac{\pi}{3})). \quad (6)$$

**Step 3.** Then, the hexagon center coordinates is accumulated to the hexagons list in order to make a search phase. To read the points under the hexagon, to add it to the points list, we need to determine the region belong to the hexagon. A rectangle with (X, Y) dimensions, where $X = 2L$ *and* $Y = 2LSin(\frac{\pi}{3})$, is created as shown in Fig.5.

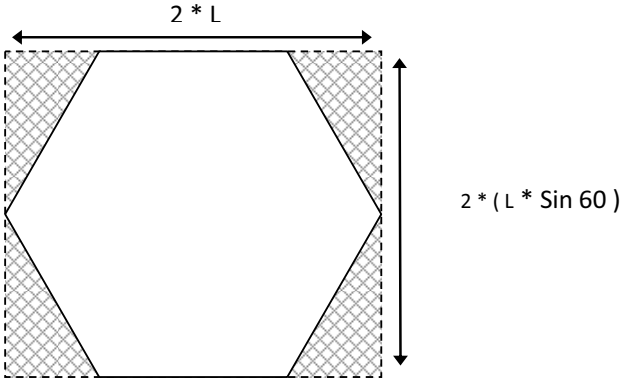


Fig. 5. Depiction of hexagon dimension.

**Step 4.** After the hexagon region is determined, then test the points beneath it for the black points to add it to the points list. Create pointers to the hexagon list to determine which hexagon the points belong and which points belong to the hexagon. Then, redo these processes every time upon creating a hexagon, using equations (7)-(10).

$$R_t = y - Lsin(\frac{\pi}{3}) \quad (7)$$
$$R_b = y + Lsin(\frac{\pi}{3}) \quad (8)$$
$$R_l = x - L \quad (9)$$
$$R_r = x + L \quad (10)$$

**Step 5.** To create the rest of the hexagon cells, the hexagon center is determined in the first level according the following equations and the entire process of finding first hexagon level is represented in Fig.6.

$$L_x = x \quad (11)$$
$$L_y = y - Level(2LSin(\frac{\pi}{3})) \quad (12)$$

The next step is constructing the other hexagons, locating the centers and then constructing the hexagons as the following:

- points from top to right

$$\sum_{i=0}^{Level} L_x = L_x + \frac{1}{2}L, L_y = L_y + LSin(\frac{\pi}{3}) \quad (13)$$

- points in the right side

$$L_x = x \quad (14)$$
$$\sum_{i=0}^{Level} L_y = L_y + 2LSin(\frac{\pi}{3}) \quad (15)$$

- points from right to bottom

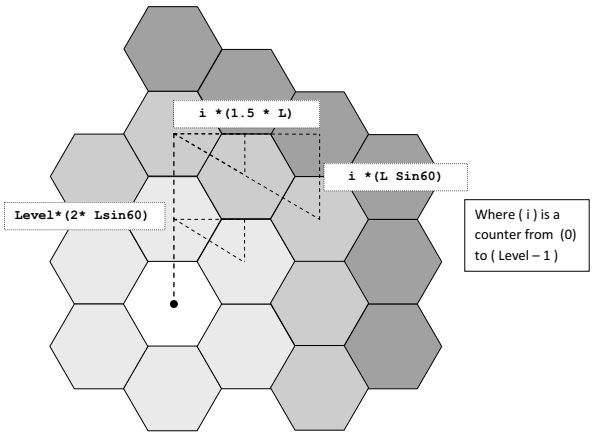


Fig. 6. Representation of first hexagon levels.

$$\sum_{i=0}^{Level} L_x = L_x - \frac{1}{2} * L, L_y = L_y + LSin(\frac{\pi}{3}) \quad (16)$$

- points from bottom to left

$$\sum_{i=0}^{Level} L_x = L_x - \frac{1}{2} * L, L_y = L_y - LSin(\frac{\pi}{3}) \quad (17)$$

- points in the left side

$$L_x = x \quad (18)$$

$$\sum_{i=0}^{Level} L_x = L_x - \frac{1}{2} * L, L_y = L_y - 2LSin(\frac{\pi}{3}) \quad (19)$$

- points from left to top

$$\sum_{i=0}^{Level} L_x = L_x - \frac{1}{2} L, L_y = L_y - LSin(\frac{\pi}{3}) \quad (20)$$

Based on the proposed method, in the search phase, a point is chosen from the collected points in 2D space. Then, the initial hexagon center point has been determined which belongs to the search space boundary. Initial hexagonal is as level 0 by calculating the location of the hexagons centers, and comparing them with other hexagons, and determine the closest other hexagons, and their levels. The goal is to specify the nearest neighbor for a given point through searching on such levels. By searching through level 0, and level 1, if there is no nearest point, then it will go to the next highest levels of the hexagonal in the space. If nearest point is found and this is a required one, then, mark the points and continue the search space until the end-of-search space is expected. Fig. 7 shows a complete searching on 2-D space.

Searching of three dimensional (3-D) space has been done by the selecting initial search points $x, y$, and $z$. The search

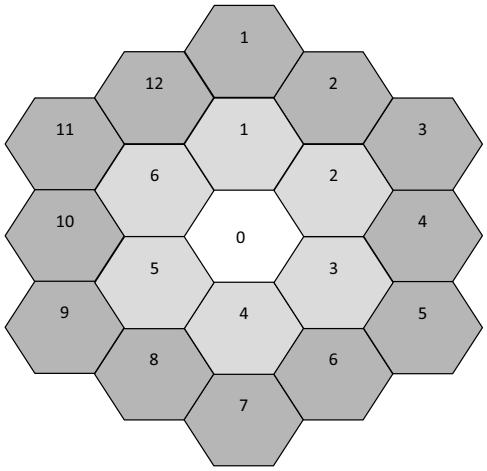


Fig. 7. A complete hexagonal searching of 2-D space.

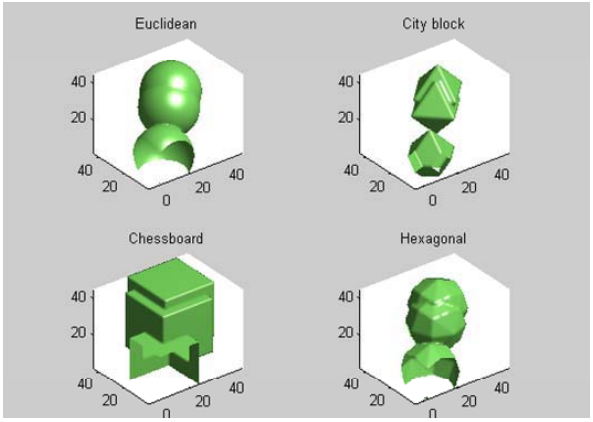


Fig. 8. A sample searching for the comparison of Euclidean, city block, chess board, and hexagonal on 3-D space.

space boundaries are fixed based on the given data set such as $I(x, y, z)$. Hexagonal transform is performed on the 3-D space. In the transform, a 3-D distance transform is started in a center of the hexagonal and process is repeated until the entire points in the space has been visited. Fig. 8 shows the comparison result of chess board, Euclidean, city block and hexagonal distances.

In the chess board distance computation method, a minimum of different center points are calculated and in the Euclidean, square root of the square of sum of the distance has been computed whereas in the city block method sum of the absolute distance is computed. The distance is computed with the condition as $\chi = |x_1 - x_2| > |y_1 - y_2| \; and \; > |z_1 - z_2|$ and Hexagonal space distance computed as in (21).

$$D_h = \begin{cases} |x_1 - x_2| + |z_1 - z_2| + \sqrt{3} - 1|y_1 - y_2|, & if \chi >= \eta \\ \sqrt{3} - 1|x_1 - x_2| + |z_1 - z_2| + |y_1 - y_2|, & \text{otherwise} \end{cases}. \quad (21)$$

Based on searching behavior of these four methods, hexagonal searching has covered an optimal regions of voxels

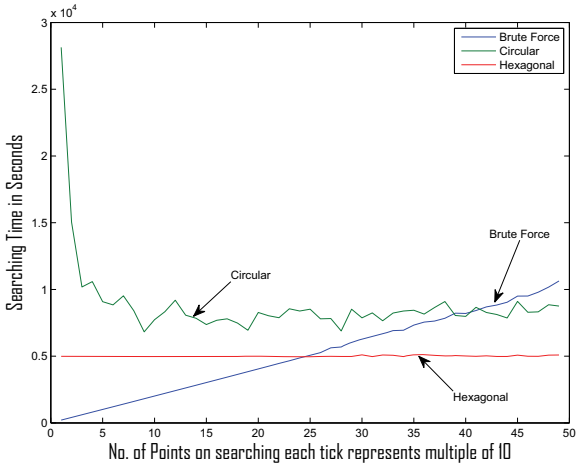


Fig. 9. Time complexity of Search points of diverse methods.

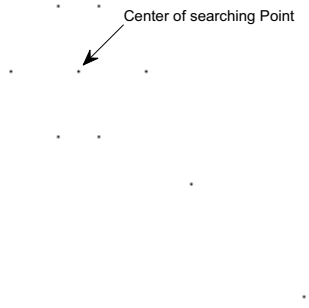


Fig. 10. A sample Hexagonal searching with three query points and illustration of one query point search set.

when an active-region-of-interest (AROI) is localized as compare to other three approaches.

## IV. Results Analysis

The proposed method using hexagonal cells has been implemented in C++ and results analysis were done in Matlab 7. Three different phases of experiments were conducted. During the first phase, for the same data set, Brute and Circular methods were tested. The time complexity of the search space was measured and recorded in the repository. In the second phase, 2-D Fourier space coefficients were searched and its distances errors were computed. Discrete cosine space has been searched using our proposed methodology in the third phase of experiments.

### A. Comparison of Searching Time Complexities

A real number space is formed for searching the points. In our experiments, multiple of 10 points were searched with three different methods such as brute force, circular and hexagonal neighbor searching methods. The turn around time for entire search space has been observed and depicted in Fig.9. It reveals that brute force searching requires around 40% less turn around time for 10 number of points whereas circular method requires around five more times as compare with hexagonal methods. However, if more points are required to search as shown in Fig. 9, around 490, then, hexagonal outperforms other two methods. The search points are a multiple of tens for each ticks on $x$-axis and zeros entries in $y$-axis mean turn around time less than one second in Fig. 9.

### B. Distance error in Fourier space

In this experiment, we have done searching of 2-D fourier coefficients based on the hexagonal approach. Since this space is a combination of real and complex numbers, in

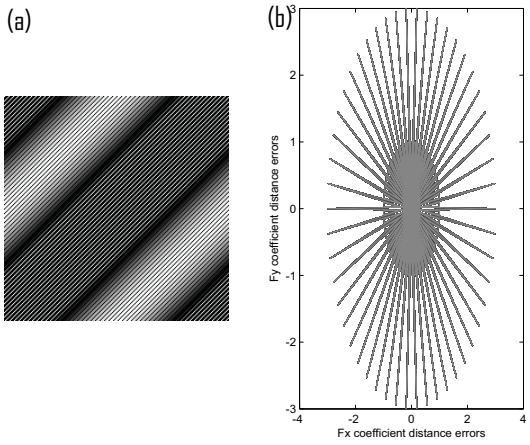


Fig. 11. 2-D Fourier space and its distance errors (a). Illustration of 2-D Fourier real coefficients (b) Error based on the query point and the distance of hexagonal computation.

that particularly locating the center point of the query point is a cumbersome process. However, hexagonal method was produced $\pm 4$ pixels distance errors among $F_x$ coefficients and $\pm 3$ pixels of distance errors in $F_x$ coefficients. A sample hexagonal searching based on the center point is illustrated in Fig. 10. This sample real-space was transformed into Fourier space and then searching was initiated to find the nearest points of the AROI.

Fig. 11a shows a 2-D Fourier space of the searching points and their distance errors based on the searching of AROI is shown in Fig. 11b.

### C. Distance error in Discrete Cosine space

Discrete Cosine Transform (DCT) is an important process of certain kinds of applications in remote sensing, compression and computer vision hence we have done the experiments

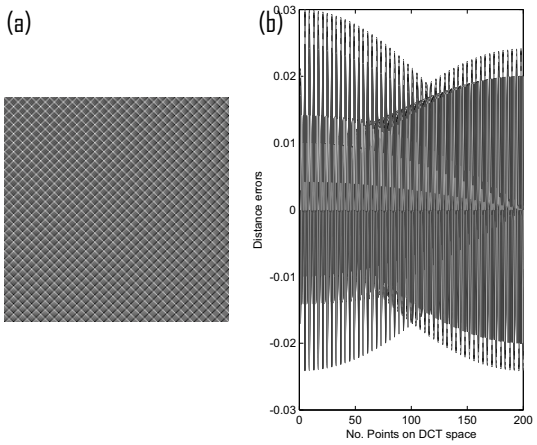


Fig. 12. Representation of DCT space and its distance errors (a). DCT space of a sample search space (b) Error based on the query point and the distance of hexagonal computation.

on this space of domain that how our proposed method will perform while seeking the query points. For that, a collection of sample data set has been transformed into Cosine frequencies and a set of query points have searched. Fig. 12a shows a sample DCT search space. 200 Cosine frequencies points have been taken for searching the query points and their distance errors were studied. The distance errors were $\pm 0.03$ ranges in the Cosine space and shown in Fig. 12b.

## V. Conclusion

The nearest neighbor searching is an important problem and deserves more attention because it has wide range of applications. In this paper, a robust searching method is proposed based on the hexagonal portrayed. The method was tested on sample sets with diverse data sizes of images. Searching of query points was done on three different spaces as real-space, Fourier space and DCT space. Based on the experimental results, we found that the Hexagonal-cells method outperforms other methods in term of turn around time.

The following conclusions can be made:

Time for both Brute method and circular methods for searching about the nearest neighbors is increasing as the number of searched point's increases.

The time needed in searching about the nearest neighbor using the Hexagonal Cells method is almost constant, for the tested samples, and it can be said that it doesn't depend strongly on number of points searched.

The time needed in searching about the nearest neighbor for searched points less than (nearly) 400 is similar in the three methods.

Future enhancement: It is recommended to create another method depending on creating another form of searching may be the pentagon. It is expected that the searching time will be reduced. Because the time for circular is more, in hexagons cells it is reduced. In pentagon it is expected to be less, i.e. as the number of sides of polygon increase the time increase, if it is known that the circle can be represented as a polygon with infinite sides.

## Acknowledgment

The author would like to thank the management, Sur University College, for providing the research support for this research study.

## References

[1] Ahmad Al-Ogaili, "Nearest Neighbor Searching using Hexagonal Cell", M.Sc. Thesis, The University of Jordan, 2010.
[2] Sunil Arya, David M. Mount, Nathan S. Netanyahu,Ruth Silverman,Angela Y. Wu,"An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions," Journal of the ACM (JACM) JACM, Vol. 45 Issue 6, pp. 891-923, Nov. 1998.
[3] J. L. Bentley, "Multidimensional binary search trees used for associative searching," Communications of the ACM, 18(9), pp. 509-517, 1975.
[4] Lawrence Cayton, Sanjoy Dasgupta,"A Learning Framework for Nearest Neighbor Search," Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007.
[5] Kenneth L. Clarkson, "Nearest-Neighbor Searching and Metric Space Dimensions," Tech. Report of Bell Labs, NJ, pp. 1-45, Oct. 2005.
[6] Andrew T. Hudak, Nicholas L. Crookston, Jeffrey S. Evans, David E. Hall, Michael J. Falkowski,"Nearest neighbor imputation of species-level, plot-scale forest structure attributes from LiDAR data," Remote Sensing of Environment, No. 112, pp.2232-2245, 2008.
[7] David Marshall, "Nearest Neighbour Searching in High Dimensional Metric Space," Master Thesis, Australian National University, 2006.
[8] Roberto Paredes and Enrique Vidal, "Learning Weighted Metrics to Minimize Nearest-Neighbor Classification Error," IEEE Trans. on Patt. Anal. Mach. Int., vol. 28, No. 7, pp.1100-1110, JULY 2006.
[9] Sun, J., Tao, Y., Papadias, D., and Kollios,G., "Spatio-temporal join selectivity," Information Systems Journal Elsevier. vol.31, no. 8 , pp. 793-813, 2006.
[10] P. Viswanath, K. Rajesh, C. Lavanya, Y.C.A. Padmanabha Reddy,"A Selective Incremental Approach for Transductive Nearest Neighbor Classification," Proc. IEEE, 978-1-4244-9477-4/11, pp. 221-226, 2011.
[11] Hui W., "Nearest Neighbors by Neighborhood Counting," IEEE TPAMI, vol. 28, no.6, pp. 942-953, June 2006.
[12] Xianfei Zhang, Bicheng Li, Xianzhu Sun, "A k-Nearest Neighbor Text Classification algorithm Based on Fuzzy Integral," Proc. of IEEE, Sixth International Conference on Natural Computation (ICNC 2010), 978-1-4244-5961-2/10, pp. 2228-2231, 2010.

**Ahmad Attiq AL-Ogaibi** Ahmad Attiq Al-Ogaibi had completed his Master degree in Computer science from the University of Jordan. His field of interest includes Pattern recognition, Image processing and Multimedia.

**Moh'd Belal Al-Zoubi** Moh'd Belal Al-Zoubi had completed his Bachelor degree in Computer science from Belgrade, Yugoslavia, 1983 and Master degree in Computer science from Detroit, USA, 1985. He completed his PhD from Leeds, UK, 1995. Currently, he is a Professor of Computer Information Systems at the University of Jordan, Jordan. He has published several research papers in the International Journals and Conferences. His field of research interest includes Pattern recognition, Image Processing, Multimedia and Information Retrieval. His specializations are Data Mining / GIS and Computer Graphics.

**Ahmad Sharieh** Ahmad Sharieh had two bachelor degrees: one in Mathematics and one in Computer Sciences. He had master degree in Computer Science and High Diploma in Teaching in Higher Education. He had PhD in Computer and Information Sciences from Florida State University 1991. Sharieh worked as Assistant Professor in Fort Valley College / USA and The University of Jordan (UJ) / Jordan. He worked as Associate Professor with Amman Arab University for Graduate Studies (AAUGS) / Jordan and The University of Jordan. He worked as Dean

of King Adbullah II School for Information Techology/Jordan. Currently, he is a professor of Computer Sciences and Dean at Sur University College (SUC), Oman. He published articles in journals (27), in conferences (22), and authored and prepared books (14). He gained grant for eight research projects from UJ and Europe. He developed several software systems such as: Teaching Sign Language, e-learning Modeling and Simulation, and Online (Automated) Exams. He is on the editorial board of several journals and conferences and a referee of several others. His research areas are Distributing Systems, Expert Systems, E-Government, E-Learning, Parallel Processing, Pattern Recognition, Software Engineering, Wire/Wireless Communication, Modeling and Simulation.

**Bremananth R** received the Bachelor and Master degrees in Computer Science from Madurai Kamaraj and Bharathidsan University in 1991 and 1993, respectively. He obtained Master of Phil. degree in Computer Science and Engineering from Government College of Technology, Bharathiar University, in 2002. He received his Ph.D. degree in 2008 from Department of Computer Science and Engineering, PSG College of Technology, Anna University, Chennai, India. He has completed his Post-doctoral Research Fellowship (PDF) from the School of Electrical and Electronic Engineering, Information Engineering (Div.) at Nanyang Technological University (NTU), Singapore, 2011. Before joining NTU, Singapore, he was a Professor and Head, Department of Computer Science and Application, in India. He has 20+ years of experience in teaching, research and software development at various Institutions. Currently, He is an Assistant Professor for Information Technology at Information Systems and Technology Department, Sur University College, Sur, Oman, affiliated to Bond University Australia. He is an associate editor of various International Journals in USA and He is an active reviewer of various IEEE International conferences/Journals. His fields of research are Acoustic holography, Acoustic imaging, Pattern recognition, Computer vision, Image processing, Biometrics, Multimedia, Computer network, Software engineering, Soft computing and Microprocessors.

Dr. Bremananth received the M N Saha Memorial award for the Best Application Oriented paper in the year 2006 by Institute of Electronics and Telecommunication Engineers (IETE). His continuous contribution of research was recognized by Who's who in the world, USA and his biography was published in the year 2006. He is a member of Indian society of Technical Education (ISTE), Advanced Computing Society (ACS), International Association of Computer Science and Information Technology (IACIT) and Institute of Electronics and Telecommunication Engineers (IETE).