International Journal of Technology Diffusion
Volume 11 • Issue 4 • October-December 2020

Localization of Autonomous Robot in an Urban Area Based on SURF Feature Extraction of Images

Abu Sadat Mohammed Yasin, Universitat Rovira i Virgili, Spain
Md. Majharul Haque, Bangladesh Bank, Bangladesh
Md. Nasim Adnan, Jashore University of Science and Technology, Bangladesh
Sonia Rahnuma, Bangladesh Bank, Bangladesh
Anowar Hossain, Brain Station 23, Bangladesh
Kallol Naha, Universitat Rovira i Virgili, Spain
Mohammod Akbar Kabir, University of Dhaka, Bangladesh
Francesc Serratosa, Universitat Rovira i Virgili, Spain

ABSTRACT

An autonomous robot is now an internationally discussed topic to ease the life of humans. Localization and movement are two rudimentary necessities of the autonomous robots before accomplishing any job. So, many researchers have proposed methods of localization using external tools like network connectivity, global positioning system (GPS), etc. However, if these tools are lost, either the movement will be paused, or the robot will be derailed from the actual mission. In these circumstances, the authors propose an approach to localize an autonomous robot in a specific area using the given set of images without external help. The image database has been prepared and kept in the internal memory of robot so that image matching can be done quickly. The localization method has been accomplished using three algorithms: (1) SURF, (2) ICP-BP, and (3) EMD. In the evaluation, SURF has been found better than ICP-BP and EMD in terms of accuracy and elapsed time. The authors believe that the proposed method will add value to other methods using some external tools even when those tools are unavailable.

KEYWORDS

Autonomous Robot, Feature Extraction, Global Positioning System, Image Database, Localization, SURF

INTRODUCTION

Geo-localization is the identification of the real-world geographic location of an object. It is closely related to geographic coordinate positioning systems such as a radar source, mobile phone, Internet connected computer terminal, autonomous robot or any kind of automatically moving objects. Internet and computer-based geo-localization can be accomplished by associating a geographic location with the Internet Protocol (IP) address, MAC address, RFID, hardware embedded article/

DOI: 10.4018/IJTD.20201001.oa1

production number, embedded software number, Wi-Fi positioning system, device fingerprint, canvas fingerprinting, device's GPS coordinates, or some self-disclosing information (Holdener & Anthony, 2011; Haque et al., 2013).

Autonomous robots, just like humans, also have the ability to make their own decisions and then perform an action accordingly. A truly autonomous robot is one that can perceive its environment, make decisions based on what it perceives and/or has been programmed to recognize and then actuate a movement or manipulation within that environment ("Autonomous Robots", 2020). Geo-localizing the current position of autonomous robot is a significant issue because the robot needs to know its current location before any movement within a reasonable time-frame.

Many prospective researchers have proposed different methods for the geo-localization of an autonomous robot using Radio Frequency (RF), GPS, Internet, laser system, ultrasonic sensor, landmarks, skylines etc. A detailed overview of these relevant papers is given below.

In 1995, Magee & Aggarwal (1995) presented a computationally straightforward method for determining location of camera that is mounted on a robot. The positioning of robot from sensor data was proposed by Burgard, Fox, and Thrun (1997). This approach provides logical criteria for (i) setting the robot's motion direction (exploration), and (ii) determining the pointing direction of sensors to efficiently localize a robot. A low-cost strategy for localization was proposed by utilizing a Kalman filter operating on sensors' data for estimating the position and orientation of robot (Goel, Roumeliotis, & Sukhatme, 1999).

Han, Lee, and Hashimoto (2000) offered an approach by using binocular stereo vision to control the position and orientation of a robot. This method works for SCARA (Selective Compliance Assembly Robot Arm or Selective Compliance Articulated Robot Arm) robot manipulator. Another localization method was proposed by Yun, Lyu, and Lee (2006) which utilize the external monitoring camera information for the indoor environment. Two methods for simultaneous localization and mapping for both outdoor and indoor environments were described by Berrabah and Bedkowski (2008). The first method (Berrabah & Bedkowski, 2008) is a feature-based algorithm that combines geo-referenced images to localize robot in user-defined global coordinates frame. The second method (Berrabah & Bedkowski, 2008) works in indoor environment and robot uses a laser range finder to build an occupancy grid map in its navigation area.

A localization method using the Matrix Pencil (MP) algorithm for hybrid detection of the Direction of Arrival (DOA) and Time of Arrival (TOA) was presented in (Trinh et al., 2012). Huang, Tsai, & Lin (2012) published two techniques for mobile robot localization for the indoor environment. At first, they (Huang et al., 2012) use the images of the markers attached on the ceiling with known positions to calculate the location and orientation of the robot. Secondly (Huang et al., 2012), an RGB-D camera mounted on the robot is adapted to acquire the color and depth of images of the environment. A real-time 3D localization and mapping approach for the USAR (Urban Search and Rescue) robotic application was proposed by Bedkowski, Maslowski, and Cubber (2012).

A system for large scale location recognition for the environment based on skyline segmentation where no GPS information is available was presented by Saurer et al. (2015). Shwe and Win (2017) developed a map-based self-localization for autonomous mobile robot navigation which can be used in a structured indoor environment only. It (Shwe & Win, 2017) is based on the visual detecting process by fusing with the ultrasonic sensor and encoder method. In 2018, Wu, Zheng, Bao, and Li (2018) published a reconfigurable micro mobile robot cluster system based on precision detection.

Indeed, a lot of research works have been accomplished on robot relocation at indoor and outdoor with the help of various technologies or systems like RF, Internet, GPS, WiFi, indication sign, other different networks, etc. (Wu, 2016). In the aforementioned research works, some (Magee & Aggarwal, 1995; Burgard et al., 1997; Yun et al., 2006; Huang et al., 2012; Shwe & Win, 2017) are only for indoor purposes and applicable in a structured environment. The method of (Yun, Lyu, & Lee, 2006) needs the help of an external camera like CCTV. Another method (Han et al., 2000) guides a SCARA robot manipulator to reach the desired location without knowing or identifying its current position

which is actually the movement of arm only. Two different methods have been presented in (Berrabah & Bedkowski, 2008) where the first method needs geo-reference images for outdoor localization and the second method uses a laser range finder to localize itself in an indoor environment. The method (Bedkowski et al., 2012) has also used a commercially available laser management system. Another method (Saurer et al., 2015) worked for the mountainous terrain area based on skyline segmentation. However, it (Saurer et al., 2015) may not be applicable in city streets (particularly old town streets) where there may have too many obstacles for visibility. Moreover, RF has been utilized in (Trinh et al., 2012) and ultrasonic sensors, encoder methods, artificial landmarks, etc. have been used in (Shwe & Win, 2017). The method (Wu et al., 2018) seemed to be for an indoor environment based on robot marking or label detection.

It has been discussed that many prospective researchers have contributed to the arena of geo-localization for the autonomous robot at in-door and out-door in flat and terrain regions with the help of RF, GPS, Internet, Skyline, ultrasonic sensor, landmarks, laser management system, etc. If none of the above helping tools are available at any moment while the robot is being moved and it is crucially needed to move, the autonomous robot may be derailed or look for an alternative way of self-localization. Considering these critical circumstances, we propose a solution in this paper where all images and coordinates for a specific area are saved in an internal database of the robot. If the robot is placed anywhere in the area, it will take a new picture and match it with the internal database to obtain its current position in a reasonable time. We believe that our method can significantly add value to other methods and open the window of an alternative way of localization in no-time.

In doing so, we build a database consisting of (i) 478 pictures of "Sagrada Familia - A church at Barcelona" (shown in Figure 1), (ii) 3D-points matrix in Matlab ".mat" format and (iii) 2D-points

Figure 1. Images of "Sagrada Familia"

matrix in Matlab ".mat" format, and (iv) camera positions matrix for every image in 3D coordinate in Matlab ".mat" format (Garrell & Sanfeliu, 2012). The database is described in section "DATABASE".

In the proposed method, the database resides in the memory of the autonomous robot. The robot has no implemented geo-localization system like GPS, Internet or any other network connectivity. The autonomous robot has only to capture the image(s) and use them as input (as shown in Figure 2). It has to look for the best matched or closest image from the database and assume its current position by matched image's position. The closest image is the one that obtains a minimum distance or best matched with the autonomous robot's input image. While designing and implementing the localization method, both the accuracy and execution time should be kept as top priority because an autonomous robot needs to know the current position spontaneously in run-time. Otherwise, it must reduce its speed which will be an obstacle in achieving the objective of an autonomous robot in every mission.

In brief, the contribution of this paper is to solve the problem of finding the current location for an autonomous robot in the area where there is no GPS, Internet, or any other network connectivity. The proposed method will also add value to the other methods that are using some external

Figure 2. Image taken by the autonomous robot

helping tools if these tools are not available somehow. Moreover, the localization method has been accomplished here using three latest algorithms and a better solution has been identified among them in terms of accuracy and elapsed time. Three algorithms are (i) Matlab Feature Extractor (Speeded Up Robust Features – SURF), (ii) Iterative Closest Point-Bipartite (ICP-BP) and (iii) Earth Mover Distance (EMD).

The rest of the paper is organized as: Description of relevant algorithms is provided in the next section. After that the constitution of the database is explained. Experiments, Evaluation and discussion on results are portrayed after database section. Finally, the concluding remarks with limitations and future works are presented.

ALGORITHMS

In this paper, our idea is to design a method where a robot will find out its location based on an input image with the help of its internal image database. The entire process will be offline and there will be no help of any third party media like GPS, Radio Frequency, laser or any other devices. Hence, we have chosen three algorithms (SURF, ICP-BP and EMD) based on computer vision, image processing, graph matching, distance calculation, etc. Point to be mentioned that (i) SURF finds out the number of matching points or similarities between two images, (ii) ICP-BP together used to find out the minimum distance or dissimilarity between two images, and (iii) EMD is also used to find out the minimum distance or dissimilarity between two images. Hence, using these three algorithms, we can find out similar or closely matched image from a set of images based on maximum matching points or similarities and minimum distances. In this regard, we have chosen these three algorithms to find out similar or closely matched image from a set of images or from internal image database. In this research work, the localization based on internal image database has been implemented by these three algorithms. The more details about these three algorithms are given as follows.

Speeded Up Robust Features - SURF

Feature extraction is a type of dimensionality reduction that efficiently represents interesting parts of an image as a compact feature vector (“Feature extraction”, 2019). This approach is useful when image sizes are large and reduced feature representation is required to quickly complete

tasks such as image matching and retrieval. Feature detection, feature extraction, and matching are often combined to solve common computer vision problems such as object detection and recognition, content-based image retrieval, face detection and recognition, texture classification, etc. ("Feature extraction", 2019).

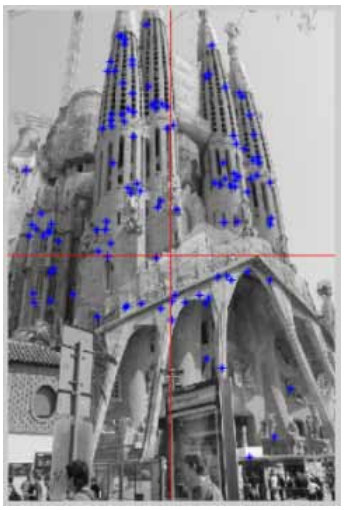
Common feature extraction techniques are Histogram of Oriented Gradients (HOG), Speeded Up Robust Features (SURF), Scale-Invariant Feature Transform (SIFT), Local Binary Patterns (LBP), Haar wavelets, and Colour Histograms. SURF is the fastest algorithm among all of these feature extraction techniques (Bay, Ess, Tuytelaars, & Gool, 2008a) and hence in our experiment, we have used the SURF feature extraction technique.

SURF algorithm is based on the Scale-Invariant Feature Transform (SIFT) algorithm. The standard version of SURF is several times faster than SIFT (Ghosh, Pandey, & Pati, 2015). SURF is a detector and a high-performance descriptor of point of interest in an image where the image is transformed into coordinates, using a technique called multi-resolution. SURF algorithm has the following steps (Bay et al., 2008b):

1. Interest Point Detection:
    a. Integral Images;
    b. Hessian Matrix-Based Interest Points;
    c. Scale Space Representation;
    d. Interest Point Localization;
2. Interest Point Description and Matching:
    a. Orientation Assignment;
    b. Descriptor based on Sum of Haar Wavelet Responses;
    c. Fast Indexing for Matching.

SURF is applied to 2D points (as shown in Figure 3) to extract features information and find out the number of matching points or similarities between two images. We extract some salient 2D points for each image and one image is selected (as shown in Figure 3) randomly from the database.

Figure 3. Salient 2D points of an image



Iterative Closest Point (ICP) and Bipartite (BP) Algorithm

1. **Iterative Closest Point (ICP):** ICP (Besl & Mckay, 1992) is an algorithm employed to minimize the difference between two clouds of points. In this algorithm, one point cloud (the reference or target) is kept fixed, while the other one (the source or input) is transformed to the best match with the reference. The algorithm iteratively revises the transformation (a combination of translation and rotation) needed to minimize the distance from the source to the reference point cloud. ICP takes reference and source point clouds to perform initial estimation of the transformation to align the source to the reference (optional) and finally sets criteria for stopping the iterations as input. After that, it transforms the source point clouds into targeted refined transformation as follows:
   a. For each point in the source point cloud, find the closest point in the reference point cloud;
   b. Estimate the combination of rotation and translation using a mean squared error cost function that will best align each source point to its match found in the previous step;
   c. Transform the source points using the obtained transformation;
   d. Iterate (re-associate the points, and so on);
2. **Bipartite Algorithm (BP):** The distance computation by the bipartite algorithm (BP) has been described here based on the research work presented in (Riesen, Neuhaus, & Bunke, 2007; Riesen & Bunke, 2009; Serratosa & Cortes, 2014). The key idea of BP is to define a dissimilarity measure for graphs or vectors. In contrast to statistical pattern recognition, where patterns are described by vectors, graphs do not offer a straightforward distance model like the euclidean distance. A common way to define the dissimilarity of two graphs or vectors is to determine the minimal amount of distortion that is needed to transform one graph into the other. These distortions are given by insertions, deletions, and substitutions of nodes and edges.

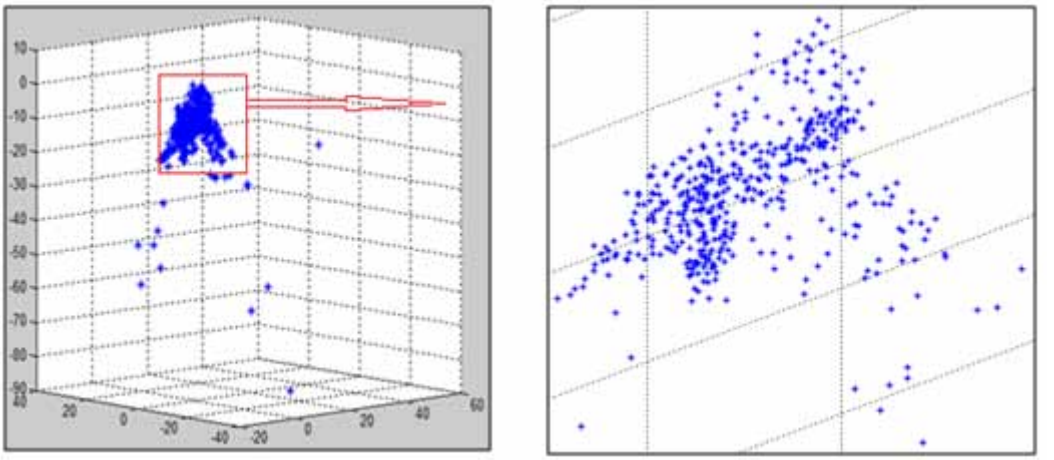
For example, there are two graphs or vectors- the source graph g_1 and the target graph g_2. The idea is to erase some nodes and edges from g_1, labeling some of the remaining nodes and edges and possibly insert some nodes and edges so that g_1 is finally transformed into g_2. A sequence of edit operations that transforms g_1 into g_2 is called an edit path between g_1 and g_2. One can introduce cost functions for each edit operation to measure the strength of the given distortion. The idea of such cost functions is that one can define whether or not an edit operation represents a strong modification of the graph. Hence, between two structurally similar graphs or vectors, there exists an inexpensive edit path that represents low-cost operations. However, for structurally different graphs or vectors, an edit path with a high cost is required. Consequently, the edit distance of two graphs or vectors is defined by the minimum cost edit path between two graphs or vectors.

In the following, a graph has been denoted by $g = (V, E, \alpha, \beta)$, where V denotes a finite set of nodes, $E \subseteq V \times V$ is a set of directed edges, $\alpha: V \rightarrow LV$ is a node labeling function assigning an attribute from LV to each node, and $\beta: E \rightarrow LE$ is an edge labeling function. The substitution of a node u by a node v is denoted by $u \rightarrow v$, the insertion of u *is presented* by $\varepsilon \rightarrow$ u, and the deletion of u is mentioned by $u \rightarrow \varepsilon$.

The edit distance can be computed by a tree search algorithm, where possible edit paths are iteratively explored and the minimum cost edit path can finally be retrieved from the search tree (Serratosa & Cortes, 2014). This method allows us to find the optimal edit path between two graphs or vectors.

We have extracted some salient 3D points for each image and the same image (as shown in Figure 3) has been selected from the database. ICP and BP together work on 3D points (as shown in Figure 4) and calculate the distance between two images.

Figure 4. Salient 3D points of image



Earth Mover's Distance (EMD)

EMD (Serratosa & Sanfeliu, 2006) is a method to evaluate dissimilarity between two multi-dimensional distributions in some feature space. For two given distributions, one can be seen as a mass of earth properly spread in space and the other as a collection of holes in the same space. EMD algorithm measures the least amount of work needed to fill the holes with the earth where a unit of work means to transport a unit of the earth for a unit of the ground hole (Rubner, Tomasi, & Guibas, 1998).

It is valid only if the two distributions have the same integral, as in normalized histograms or probability density functions (Benois-Pineau, Precioso, Cord, 2012). The EMD distance (ordinal distance) (Serratosa & Sanfeliu, 2006) between two histograms is the minimum of work needed to transform from one histogram to the other. Histogram *H(A)* of the input image can be transformed into histogram *H(B)* of the reference image by moving the elements to left or right and the total of all necessary minimum movements is the distance between them.

The distance between two histograms is defined as follows (see Equation 1, (Serratosa & Sanfeliu, 2006)):

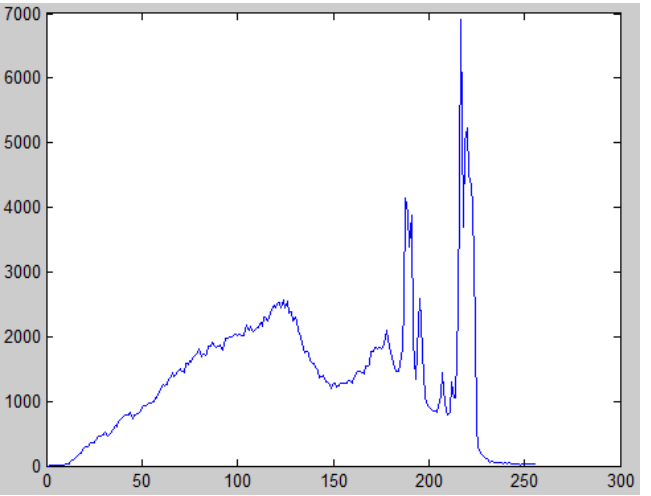
$$D_{ord}\left(H\left(A\right),H\left(B\right)\right)=\sum_{i=1}^{T-1}\left|\sum_{j=1}^{i}\left(H_j\left(A\right)-H_j\left(B\right)\right)\right| \quad (1)$$

Here, D_{ord} implies the distance between two histogram *H(A)* and *H(B)*, *T* implies the length of *H(A)*. Note that the length of both histograms has to be the same.

We convert all the colored images of "Sagrada Familia" (mentioned in Section "DATABASE") into grayscale images and then calculate their histogram. The histogram of the image (Figure 3) is presented in Figure 5. After that, EMD has been applied to find out the distance between the source and reference images.

After the implementation of three algorithms (i) SURF, (ii) ICP-BP and (iii) EMD, we calculate the accuracy in terms of neighbor images' position and time elapsed to obtain the output. The implementation detail is described in section "EXPERIMENTAL SETUP".

Figure 5. Histogram of the gray scale image



DATABASE

A sequence of 2D-pictures (two rounds) has been taken of the "Sagrada Familia" church in Barcelona, Spain, for creating the database. The first round pictures (total 364) have been captured around the church with an increase of one-degree interval concerning the center of the church and the second round pictures (total 114) have been taken with a three-degree interval. In this way, two rounds of the 360-degree view have been covered with 478 (364+114) pictures.

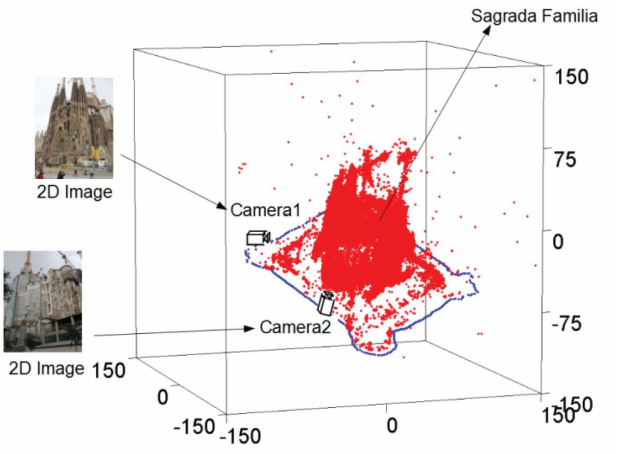
It is noticeable that "Sagrada Familia" church in Barcelona, Spain is just an example database for our experiment what we have used in this research work. Anybody can view the panorama image of this church at Visita Virtual ("Visita Virtual" 2020). The same experiment can be done for different Spots or Locations by outside or in house image database.

From the entire sequence (478 pictures), a total of 100,532 3D-points of the church have been extracted (using the Bundler method (Snavely & Todorovic, 2011)) and gathered the information of corresponding 2D-pictures which visualize these 3D points. It is worth mentioning that each picture contains from 4,000 to 40,000 3D-points. Moreover, the method (Snavely & Todorovic, 2011) returns the relation between the 3D-points and the position in pixels of the pictures. It is noticeable that the positions of the cameras were deduced by the pose estimation method (Rubio et al., 2015).

We now have a 3x100,532 matrix of 3D-points in Matlab ".mat" format, where (i) 3 represents the 3D coordinates *(u, v, w)* and (ii) 100,532 represents the total number of 3D-points (points of the church). Another matrix with size 102x3x100, 532 saved in Matlab ".mat" format, where (i) 102 represents total number (maximum) of images can be displayed by the 3D points of the church, (ii) 3 represent picture number and 2D coordinates *(i, x, y)*, where *i* value varies from 1 to 478 (total no of pictures), and (iii) 100,532 represents the total number of 3D points. We also have another matrix of camera positions with size 478x3, where (i) 478 represents the total number of images and (ii) 3 represents the camera's (which is used to capture these 478 images) 3D coordinates (*u, v, w*).

Figure 6 shows the 3D model (red points) of "Sagrada Familia", and the different poses of the camera that captured the images of the model (blue points). Axes are expressed in meters and the center of the church is the origin of the coordinate system. Note that some noisy points are remaining in the sky.

Figure 6. 3D model of "Sagrada Familia"



EXPERIMENTAL SETUP

In this section, the experiment with the three latest algorithms for the localization problem is presented and a better solution is identified. The three algorithms (SURF, ICP-BP, and EMD) have already been discussed in the second section. The performance of these algorithms has been measured based on accuracy and elapsed time. Point to be mentioned that, a database (discussed in Database section) of a total of 478 images and 100,532 3D-points have been used here for the purpose of evaluation. The entire experiment has been accomplished in five steps. Some preprocessing for experimental preparation has been performed in steps one and two. In step three, four and five have generated three matrices (with two versions) are generated by SURF, ICP-BP, and EMD algorithms respectively. SURF generates two versions of similarity matrices and other algorithms generate two versions of distance matrices. These five steps of experiments are presented as follows:

Step 1: This is a preliminary step to carry out some necessary preprocessing related tasks. At first, the 478 images are converted from JPEG format (JPEG is usually known as JPG. It stands for Joint Photographic Expert Group) into PGM (Portable Gray Map) format by using IrfanView ("Conversion of JPEG format images into PGM format", 2020). The conversion is required because SURF works on grayscale images only and EMD performs faster in the grayscale image. The 3D-points (extracted from the mentioned images and explained in database section earlier) database is a matrix with size 3x100532 which is too large to perform calculation within a rational time-frame. In addition to these 3D-points database, there are a lot of unnecessary or unwanted noise points. We need to reduce these noise points and find out only the salient points. Algorithm 1 has been used in step one to get a matrix (100532x1) of salient points in Matlab ".mat" format where 100532 rows are specific for the 100532 3D-points. In this matrix, each cell has value 1 if the point is salient otherwise the cell has value 0.

Step 2: This step is also related to preprocessing. For every image, the individual matrix is generated and save them in Matlab ".mat" format. These matrices contain all the images' information for every salient point equivalent to 2D, 3D points and their feature extraction values. Each image matrix has size n x 72 where n varies from 0 to 100532 and the columns of the matrix contain values according to Table 1. Algorithm 2 is used to generate the above-mentioned matrices (as described in Table 1).

Algorithm 1. Get salient points Matrix

```
max = 0.6 (this the arbitrarily chosen threshold point)
salients = array of Matlab ones with size (length/number of 3D-ponits)
// explained in the third paragraph of section-Database.
For i=1 to all 3D-ponits
      for j=i+1 to all 3D-ponits
            if(Euclidean norm(3D-points(i)) - Euclidean norm(3D-points(j))<
               max)
                salients(j) = 0;
            end of if
      end of for
end of for
save salient points matrix in Matlab ".mat" format
```

Table 1. Individual image Matrix

Column	Values/Information
1	Every row's 1st column is the identifier/no of 3D-points from1~100532
2 - 3	2nd and 3rd columns are 2D equivalent (x, y) coordinate values
4 - 6	4th, 5th and 6th positions are the 3D equivalent (u, v, w) coordinates
7 - 8	7th and 8th are the 2D equivalent coordinate (x', y') with respect to image size where image's centre pixel is the (0,0) position of the X and Y axis
9- 72	in the positions 9 to 72 are the Matlab feature extraction values (64 values)

Algorithm 2. Generate the image information Matrices

```
matrix p // declaration of a blank matrix;
for i = every images
for j = every 2D points // explained in the third paragraph of section-
Database.
            if j is a salient point
get the 2D (x, y), 3D (u, v, w), 2D (x', y') and SURF feature extraction
points.
add them in a new row in the matrix p.
            end of if
      end of for
end of for
save the matrix named after the image name in Matlab ".mat" format
```

Step 3: In this step, we perform the first experiment with the SURF algorithm. In this experiment, we use odd-numbered images (total 239) as the input images and even-numbered images (total 239) as reference/database images from a total of 478 images. We calculate the number of matching points between two images and generate a matrix based on the SURF algorithm. In this experiment, two versions of matrices are prepared as follows:

a. **Similarity matrix version-1:** We generate the first version matrix with size 239x288 where 239 rows represent output according to the 239 input images and the columns contain values according to Table 2. Algorithm 3 is used to generate the matrix described in Table 2;
b. **Similarity matrix version-2:** The SURF algorithm is used again to generate the second version of the matrix where we have considered only one maximum (best) match. The size of the matrix is 239 x 246 where 239 rows represent 239 outputs by the number of input images and the columns contain values according to Table 3. Algorithm 4 has been utilized to generate the matrix described in Table 3.

Step 4: In this step, we perform the second experiment with ICP and BP algorithms. It takes the same input and reference images as step three. The distances between the two images are calculated here. It is worth to mention that, we use the same approach (Algorithm 4) of Step three but instead of SURF, we use ICP and BP (explained in the Algorithm section) and consider the minimum distances. To calculate distances, both ICP and BP distance calculation algorithms are used. ICP iteratively revises the transformation (a combination of translation and rotation) to minimize the distance from the source to the reference image position (considering all the salient 3D coordinates) and then BP is used to calculate the distance between input and reference images. In this step, it also generates two versions of matrices like step three as follows:

a. **Distance matrix version-1:** The first version of the matrix has been generated with size 239x288, where 239 rows represent 239 outputs by the number of input images and the 288 columns contain values according to Table 4;
b. **Distance matrix version-2:** The second version of the matrix is generated with the same approach of step three. We use ICP and BP to calculate the distances and consider only one minimum distance. The size of the second version matrix is 239x246 where 239 rows represent outputs by the 239 input images and the 246 columns contain values according to Table 5.

Step 5: This step also takes the same inputs and generates two distance matrices (similar to Steps three and four). In this step, to calculate distances we use EMD (explained in Algorithm section). The approach of this step is the same as step four; however, image histograms of both input images and reference images as inputs and references are used. We conduct two versions for this experiment (like step three and four) and save into two distance matrices with size 239x288 and 239x246. Both of these matrices contain values according to Table 4 and Table 5 respectively.

EVALUATION AND DISCUSSION ON RESULTS

The implementation detail, of the proposed method for three algorithms, has been depicted in the experimental setup section. For evaluation, three matrices are generated from the experimentation where each matrix has two versions. The matrices are as follows:

1. Maximum Matching Points or Similarities Matrix by SURF (see Table 2 for the first version of matrix definition and see Table 3 for the second version of matrix definition);
2. Minimum Distance Matrix by ICP-BP (Table 4 for the first version matrix definition and Table 5 for the second version matrix definition);
3. Minimum Distance Matrix by EMD (Table 4 for the first version matrix definition and Table 5 for the second version matrix definition).

It is explained in the first version of matrices that there are four types of averages (mentioned in columns 264, 269, 274, and 279 of Table 2 and 4). The best average (in terms of maximum matching points or minimum distances) has been found out for each row and saved in column 285 (please see Table 2 and 4, column 285 for the definition). For all the rows of three matrices (stated in the above

Table 2. The similarity matrix (version-1) generated by SURF

Column	Values/Information
1 - 239	Contains the number of matched points or similar points between input image and reference images.
240	Contains the maximum or highest number of matching point or similar point or value from column 1 to 239
241	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose number of matching points or similarities with input image is maximum or equal to the values of column 240.
242	Contains the label according to the reference (column 241) image.
243 - 245	Contains the reference image (column 241) camera position u, v, w values.
246	Contains the second maximum or highest number of matching points or similar points or values from column 1 to 239.
247	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose number of matching points or similarities with input image is second maximum or equal to the values of column 246.
248	Contains the label according to the reference (column 247) image.
249 - 251	Contains this reference image (column 247) camera position u, v, w values.
252	Contains the third maximum or highest number of matching points or similar points or values from column 1 to 239.
253	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose number of matching points or similarities with input image is third maximum or equal to the values of column 252.
254	Contains the label according to the reference (column 253) image.
255 - 257	Contains this reference image (column 253) camera position u, v, w values.
258	Contains the fourth maximum or highest number of matching points or similar points or values from column 1 to 239.
259	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose number of matching points or similarities with input image is fourth maximum or equal to the values of column 258.
260	Contains the label according to the reference (column 259) image.
261 - 263	Contains this reference image (column 260) camera position u, v, w values.
264	Contains the first maximum matching point or similarity points, its equal to the value of column 240.
265 - 267	Contains the 3D positions of the first maximum matching point, this is equal to column 243,244,245 respectively.
268	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and average 3D positions (Columns 265,266 and 267).
269	Contains the average between first two maximum matching points or similarity points, its equal to the value of column (240 + 246)/2.
270 - 272	Contains the average 3D positions of the first and second maximum matching points, these are equal to the value of column (243+249)/2, column (244+250)/2 and column (245+251)/2.
273	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and the average 3D positions (Column 270, 271 and 272).
274	Contains the average among the first three maximum matching points or similarity points, its equal to the value of column (240 + 246 + 252)/3.
275 - 277	Contains the average 3D positions of the first three maximum matching points, these are equal to the value of column (243+249+255)/3, column (244+250+256)/3 and column (245+251+257)/3.
278	Contains the error or Euclidian distance among the input images camera position (3D coordinates) and average 3D positions (Column 275, 276 and 277).
279	Contains the average between first four maximum matching points or similarity points, its equal to the value of column (240 + 246 + 252 + 258)/4.
280 - 282	Contains the average 3D positions of the first four maximum matching points, these are equal to the value of column (243+249+255+261)/4, column (244+250+256+262)/4 and column (245+251+257+263)/4.
283	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and average 3D positions (Column 280, 281 and 282).
284	Contains the minimum error among these four errors or Euclidian distances (among the columns 268, 273, 278, 283).
285	Contains which average is the best in terms of minimum error (it is noticeable that we have calculated total 4 averages from where the best one is selected here).
286 - 288	Contains the position (3D coordinates) of this best average.

Algorithm 3. Generating similarity Matrix version-1 by SURF

```
Martix Max_Match_Matrix // declaration of a blank matrix
For all input images (odd numbered images from 1-478)
   Read the input images camera position from the camera position matrix.
   // explained in the second paragraph of section-Database.
   Read the input images matrix //explained in the second paragraph of
   section-Database.
   Matches array // declaration of an array.
   For all reference images (even numbered images from 1-478)
      Read the reference images matrices
      //explained in the second paragraph of section-Experimental Setup step
      two.
      Read feature extraction values from input images matrix.
      Read feature extraction values from reference image matrix.
      Get the number of matching using these two feature extracted values by
      using Matlab SURF
      Add this number of matching point in the array Matches
   End of for

   Find out the 1st to 4th maximum matching value.
   Find the 1st to 4th maximum matching point images' identifier from (1-478)
   Find the 1st to 4th maximum matching point images' index or label
   Find the 3D position of the first, second, third and fourth maximum
   matching point image.
   Find out the first max matching values.
   Find out the position of the first max matching values.
   Find the error or Euclidian distance for this position.

   Find out average of two max matching values.
   Find the average position of the two max matching values.
   Find the error or Euclidian distance for this position.

   Find out average of three max matching values.
   Find the average position of the three max matching values.
   Find the error or Euclidian distance for this position.

   Find out average of four max matching values.
   Find the average position of the four max matching values.

   Find the error or Euclidian distance for this position.

   Find the minimum error among these four errors.
   Find which average is the best with minimum error (values from 1 to 4)
   Find this best average matching points average 3D position.

   add all of these above find out values into a new row of matrix
   Max_Match_Matrix
End of for
Save the matrix in Matlab ".mat" format.
```

Table 3. The similarity Matrix (version-2) generated by SURF

Column	Values/Information
1 - 239	Contains the number of matched points or similar points between input image and reference images.
240	Contains the maximum or highest number of matching point or similar point or value from column 1 to 239
241	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose number of matching points or similarities with input image is maximum or equal to the values of column 240.
242	Contains the label according to the reference (column 241) image
243 - 245	Contains the reference images (column 241) camera position u, v, w values.
246	Contains the error or Euclidian distance between the input images 3D coordinates of camera position and the maximum matched reference images 3D coordinates

Algorithm 4. Generating similarity Matrix version-2 by SURF

```
Martix Max_Match_Matrix // declaration of a blank matrix
For all input images (odd numbered images from 1-478)
   Read the input images camera position from the camera position matrix.
   // explained in the second paragraph of section-Database.
   Read the input images matrix //explained in the second paragraph of
   section-Database.
   Matches array // array declaration
   For all reference images (even numbered images from 1-478)
       Read the reference images matrices//images matrices have been explained
       in the second paragraph of section-Experimental Setup step two.
       Read feature extraction values from input images matrix.
       Read feature extraction values from reference images matrices.
       get the number of matching using these two feature extracted values by
       using Matlab SURF
       add this number of matching point in the array Matches
   End of for

   Find out the maximum number of matching values.
   Find the maximum number of points images identifier from (1-478)
   Find the maximum number of points images index or label
   Find the error or Euclidian distance between input image camera position
   and resulted maximum number of point's images camera position.
   Add all of these values into a new row of matrix Max_Match_Matrix
End of for
Save the matrix in Matlab ".mat" format.
```

Table 4. The distance Matrix (version-1) generated by ICP-BP

Column	Values/Information
1 - 239	Contains the distance values between input image and reference images.
240	Contains the minimum distance value from column 1 to 239
241	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose distance with input image is minimum or equal to the values of column 240.
242	Contains the label according to the reference (value of column 241) image
243 - 245	Contains this reference image (column 241) camera position u, v, w values.
246	Contains the second minimum distance value from column 1 to 239.
247	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose distance with input image is second minimum or equal to the value of column 246.
248	Contains the label according to the reference (value of column 247) image.
249 - 251	Contains this reference image (column 247) camera position u, v, w values.
252	Contains the third minimum distance value from column 1 to 239.
253	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose distance with input image is third minimum or equal to the values of column 252.
254	Contains the label according to the reference (value of column 253) image.
255 - 257	Contains this reference image (column 253) camera position u, v, w values.
258	Contains the fourth minimum distance value from column 1 to 239.
259	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose distance with input image is fourth minimum or equal to the values of column 258.
260	Contains the label according to the reference (column 259) image.
261 - 263	Contains this reference image (value of column 260) camera position u, v, w values.
264	Contains the first minimum distance, its equal to (value of column 240).
265 - 267	Contains the 3D positions of the first minimum distance, these are equal to the value of column 243, 244 and 245 respectively.
268	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and average 3D positions (value of column 265,266 and 267).
269	Contains the average between first two minimum distances, its equal to the value of column (240 + 246)/2.
270 - 272	Contains the average 3D positions of the first and second minimum distances, these are equal to the value of column (243+249)/2, column (244+250)/2 and column (245+251)/2
273	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and average 3D positions (value of column 270, 271 and 272).
274	Contains the average between first three minimum distances, its equal to the value of column (240 + 246 + 252)/3.
275 - 277	Contains the average 3D positions of the first three minimum distances, these are equal to the value of column (243+249+255)/3, column (244+250+256)/3 and column (245+251+257)/3
278	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and average 3D positions (value of column 275, 276 and 277).
279	Contains the average between first four minimum distances, its equal to the value of column (240 + 246 + 252 + 258)/4.
280 - 282	Contains the average 3D positions of the first four minimum distances, these are equal to the value of column (243+249+255+261)/4, column (244+250+256+262)/4 and column (245+251+257+263)/4
283	Contains the error or Euclidian distance between the input images camera position (3D coordinates) and average 3D positions (value of Column 280, 281 and 282).
284	Contains the minimum error between these four error or Euclidian distance (among the value of column 268, 273, 278, 283)
285	Contains which average is the best in terms of minimum error (it is noticeable that we have calculated total 4 averages from where the best one is selected here).
286 - 288	Contains the position (3D coordinates) of this best average.

Table 5. The distance Matrix (version-2) generated by ICP-BP

Column	Values/Information
1 - 239	Contains the number of distances between input image and reference images.
240	Contains the distance value from column 1 to 239
241	Contains the identifier/index number (from 1-478, even numbers only) of the reference image whose distance with input image is minimum or equal to the values of column 240.
242	Contains the label according to the reference (column 241) image.
243 - 245	Contains the reference image (column 241) camera position u, v, w values.
246	Contains the error or Euclidian distance between the input images 3D coordinates of camera position and the maximum matched reference images 3D coordinates

points a, b, and c), we have counted the values of column 285 for different types of averages separately. All of the counted best averages have been shown in Table 6.

We have given priorities higher from 1 to 4 (SL# in Table 6). Before getting the practical results we have assumed that the average of 1st and 2nd best (2nd row of Table 6) will be the

Table 6. Different types of averages from the version-1 Matrices

SL#	Types of Averages	Number of Best Averages		
		SURF	ICP-BP	EMD
1	1st best (without any average or taking only one best result)	117	112	111
2	Average of 1st and 2nd best	98	67	69
3	Average of 1st, 2nd and 3rd best	17	21	29
4	Average of 1st, 2nd, 3rd and 4th best	12	34	30

maximum counted value. Nevertheless, the result practically disappointed us and that is why the second version of matrices has been implemented for every algorithm. From Table 6, we can deduce that whatever algorithm will be used, the average of any number of results (maximum matches or minimum distances) will not give a better outcome. So, we need to consider one best match or minimum distance.

There can be a question as "why do not average positions provide better results all the time?". It has been found from the experiment that if the input image is situated between two maximum matched or minimum distanced images (Figure 7) then these average techniques provide good results otherwise (Figure 8) these techniques do not provide good results. It is noticeable that most of the input images do not situate between two or more than two maximum matched or minimum distanced images.

In Figures 9, 10 and 11, we have displayed all the three matrices (version-2) of three algorithms (SURF, ICP-BP, and EMD respectively), rows as the input image and columns as reference image 1 to 239. In the graphical view, we have plotted these matrices values in the range (0 to 1) but actually, the matrices values are very bigger than this range (0 to 1). To convert these three matrices values in this range (0 to 1), we have reduced the cells' values of matrices by dividing the means of corresponding matrices (Equation 2):

Figure 7. Input image in the middle of two maximum matched or minimum distanced images

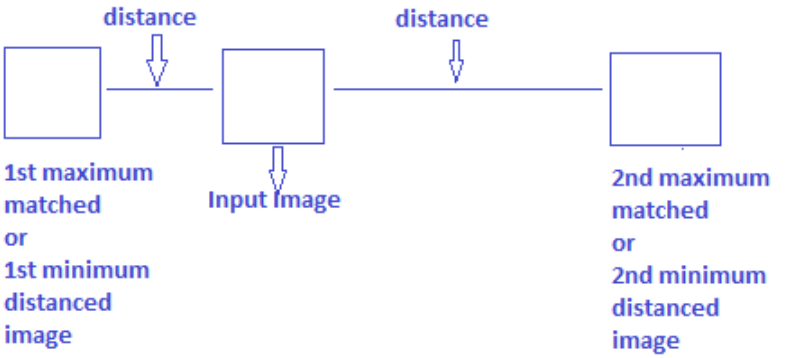


Figure 8. Input image beside the two maximum matched or minimum distanced images

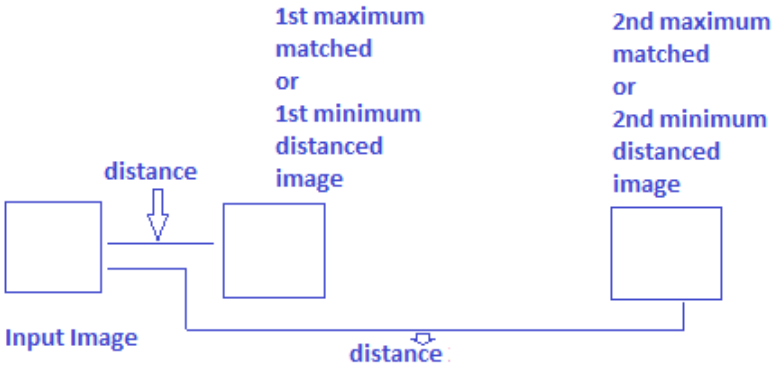


Figure 9. Graphical view of (Column 1 to 239) matching points or similarities Matrix (version-2) by SURF

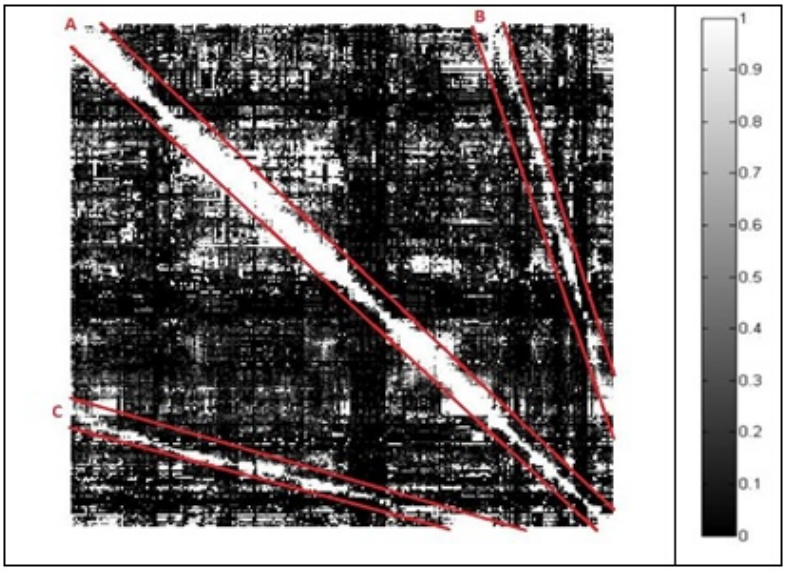


Figure 10. Graphical view of (Column 1 to 239) distance Matrix (version-2) by ICP-BP

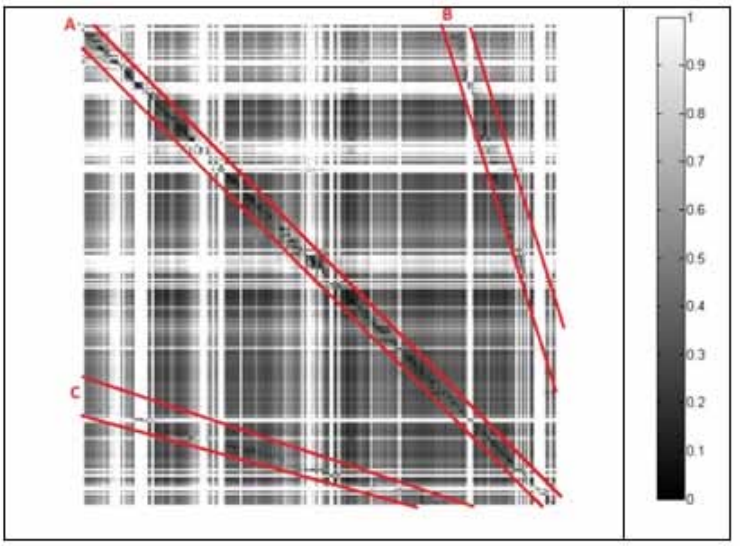
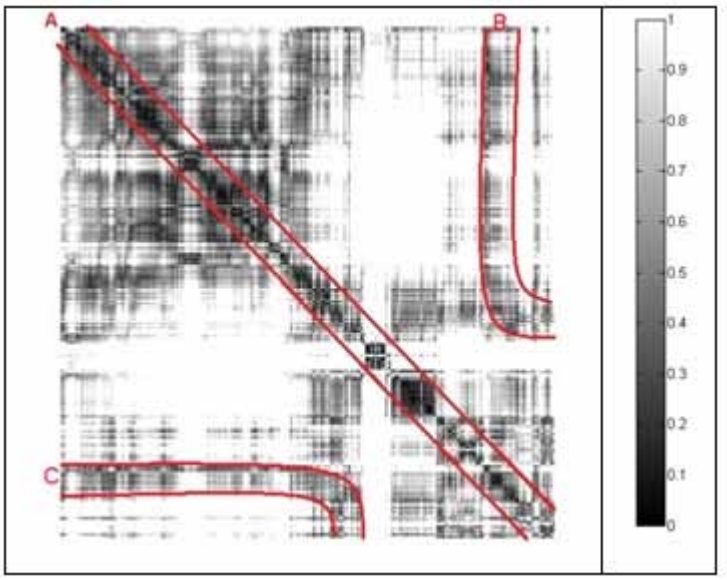


Figure 11. Graphical view of (Column 1 to 239) distance Matrix (version-2) by EMD



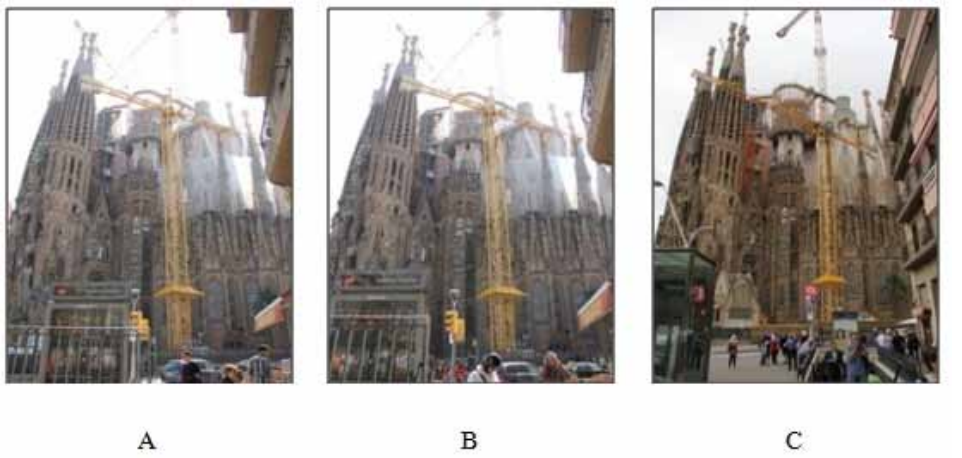
$$Reduced\ \ matrix = \frac{matrix}{mean\Big(mean\Big(matrix\Big)\Big)} \tag{2}$$

In Figures 9, 10 and 11, white color represents the maximum values and black is for the minimum values.

From these figures, we can see that every image has diagonal marking (red color line) with the label "A". In Figure 9, the diagonal with the label "A" represents the maximum matching points between 239 input and 239 reference images. In Figures 10 and 11, the diagonal with the label "A" represents the minimum distances between 239 input and 239 reference images. We have used all the odd-numbered images as input and all the even-numbered images as a reference from the "Sagrada Familia" database (Step Three of sub-section (i) under Experimental Setup section).

Besides the label "A", there are two more interesting lines labeled as "B" and "C" that can be seen in Figures 9, 10, and 11. It has already been mentioned in section "Algorithm" that we have taken a total of 478 pictures of the "Sagrada Familia" church in Barcelona (Spain), and among them, there are two rounds of 2D-pictures available. These two lines ("B" and "C") appeared because of these second-round pictures of "Sagrada Familia" where every input or source image has two very close matching reference images. For example, if we arbitrarily take 101st image out of 478 images (Figure 12(A)) as input then we will get two very closely matched images from the database and they are100th image.

Figure 12. For input image A there are two close images B and C from database



(Figure 12(B), image from the round-1) and 409th image (Figure 12(C), image from round-2). By observing these two reference images from Figure 12(B) and 12(C), we can see that they are very close or nearly the same to Figure 12(A).

In Figure 13, visualization has been given clearly for capturing two rounds of images (364 for the first round and 114 for the second round) and the positions of Figure 12 (A), 12 (B), 12 (C).

For these figures, Figure 12 (A), 12 (B), and 12 (C), we have plotted three Figures 14, 15, and 16. Indeed, these three figures are displaying one row of every matrix second version (Table 3 and Table 5). Figure 12 represents the matches or similarities between input image Figure 12(A) and all reference images (239 even-numbered images out of 478 images) calculated by SURF where we can see there are two pick values marked with red circles. These two picks represent the above examples of two reference images Figures 12(B) and 12(C). It has already been mentioned earlier that Figures 12(A), 12(B), and 12(C) are the 101st, 100th, and 409th images respectively from the database of a total of 478 images.

Figure 17 represents the distances between input image Figure 12(A) and all reference images calculated by ICP-BP. If we zoom out Figure 15(A), we can see there are two minimum values in

Figure 13. Visualization of input and reference images

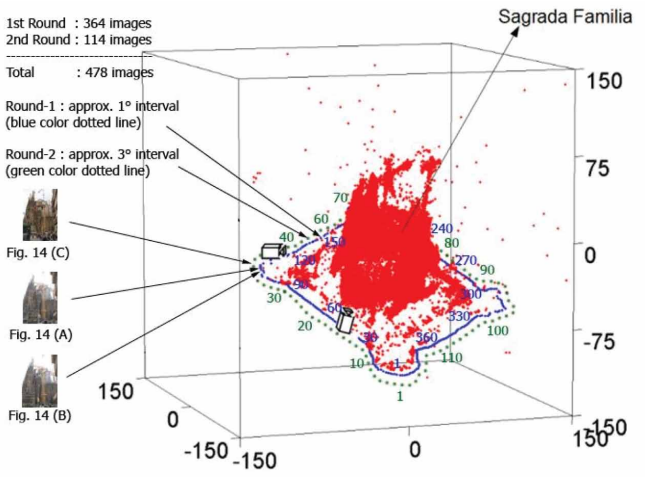


Figure 14. Graph for the 101st input image versus all the reference images of matching points or similarities Matrix (version-2) by SURF

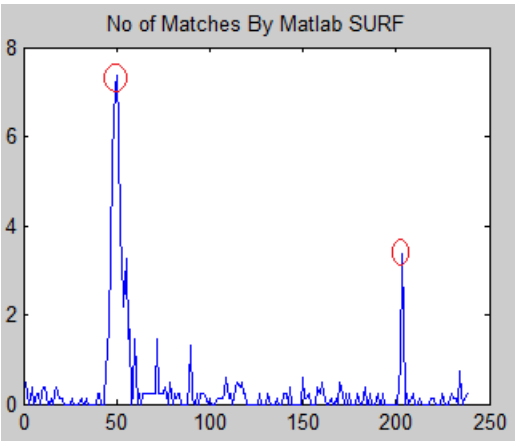


Figure 15. Graph for the 101st input image versus all the reference images of distance Matrix (version-2) by ICP-BP

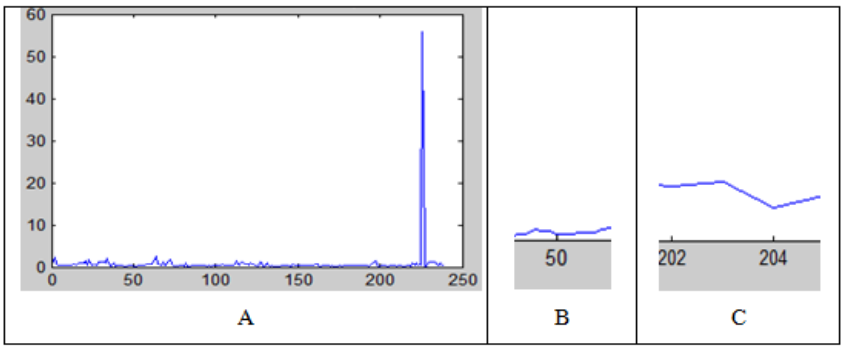


Figure 16. Graph for the 101st input image and all the reference images of distance Matrix (version-2) by EMD

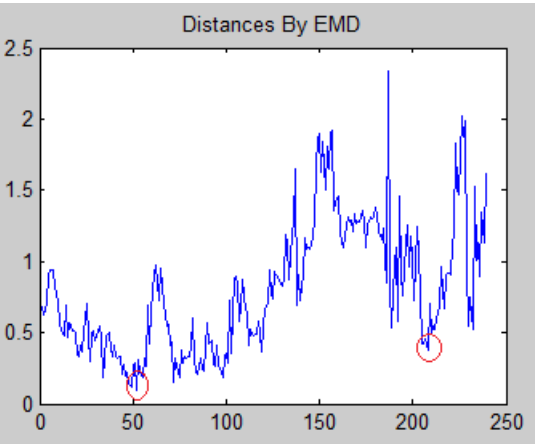
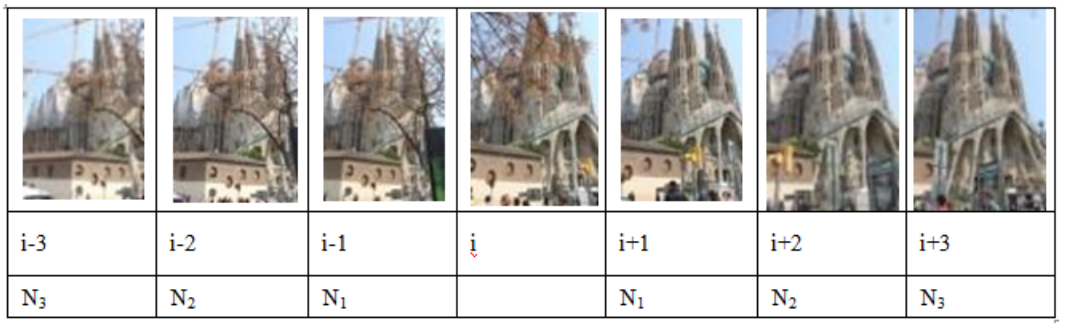


Figure 17. Neighbour images of image "i"



two separate positions like Figures 15(B) and 15(C). These two minimum values (15(B) and 15(C), part of 15(A)) represent the two reference images of Figure 12(B) and 12(C).

Figure 16 represents the distances between input image Figure 12(A) and all reference images calculated by EMD. If we closely observe this Figure 16, we can see there are two minimum values representing the two reference images Figure 12(B) and 12(C).

In the second version of every matrix, we have only taken the best maximum matching point or minimum distance value and then calculated their accuracy in terms of neighbor image (Figure 17). Neighbor image means the next and previous images of the input image from the same database. In our experiment, from the 478 images of "Sagrada Familia", for any image *i* the neighbor images are "neighbor level one, N_1 =*i+1 and i-1*", "neighbor level two, N_2 = *i+2* and *i-2*" and "neighbor level three, N_3 = *i+3* and *i-3*" (Figure 17).

It is noticeable that we already know all the images (both the 239 input/test and 239 reference/database images) position and their neighbors. The explanation of both input image and reference images have been given in section Algorithm and step three of section Experimental Setup. For all the input images, it has been checked that the resulted image is at the neighbor position or not. We have counted the total number of the resulted or output images if it is at the neighbor position N1 and N1+N2. The counted values have been presented in Table 7. According to the calculation, it has been found that SURF provides the best result among them in terms of accuracy. Point to be mentioned that, the accuracy has been measured using MATLAB.

Table 7. Accuracy in terms of neighbor reference images

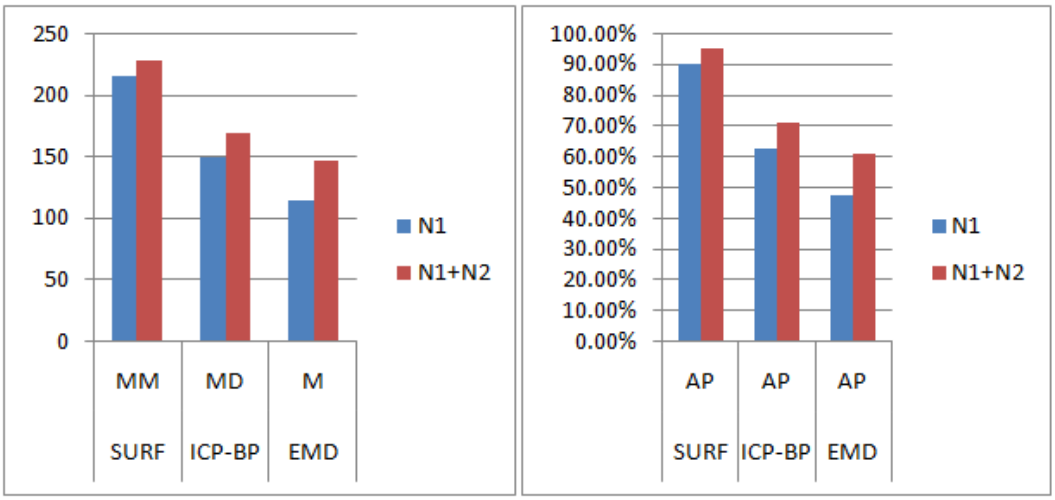
NL	SURF		ICP-BP		EMD	
	MM	AP	MD	AP	M	AP
N_1	215	89.96%	149	62.34%	114	47.70%
N_1+N_2	227	94.98%	169	70.71%	146	61.09%

Here in Table 7, NL means "Neighbor Level", MM means "Max. Matches Found out of 239 images", AP means "Accuracy in percentage", MD means "Min. Distance Found out of 239 images".

From the accuracy measurement (Table 7 and Figure 18), it can be derived that for N_1 position SURF is 27.62% and 42.26% more accurate and for N_1+N_2 position it is 24.27% and 33.89% more accurate than ICP-BP and EMD respectively.

It has already been discussed in section Experimental Setup that EMD works with an image histogram where there is a possibility of data loss. ICP-BP works on 3D points. We have almost 100,532 numbers of 3D points but due to time complexity here in this experiment, some 3D points have been used as salient points for ICP-BP. While choosing salient points, some data losses could occur or insignificant points could be specified as salient points. On the other hand, SURF works on image feature extraction values where the possibility of data loss is comparatively lower than EMD and ICP-BP. Thus, the accuracy of SURF is better than EMD and ICP-BP (Table 7 and Figure 18).

Figure 18. Accuracy in terms of neighbor reference images in graphical format

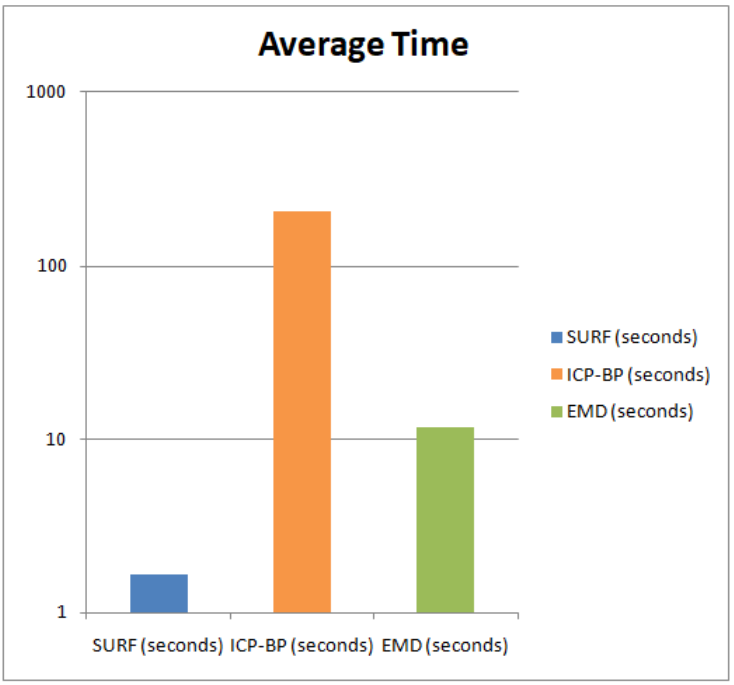


We have also calculated the time elapsed by each algorithm by using MATLAB "tic" and "toc" ("Matlab function", n.d.; "Matlab function", n.d.), hence for the first ten inputs, we have found the following elapsed times in seconds (Table 8 and Figure 19). From Table 8 and Figure 19, we can see that SURF is faster than others. It is noticeable that the algorithms have been implemented, tested and elapsed times are calculated in a laptop with Processor: Intel(R) Core(TM) i5-3210M CPU @2.50GHz 2.50GHz, Installed memory (RAM): 8.00 GB (7.60 GB usable), System type: Windows 7 Professional 64-bit Operating System.

Table 8. Time elapsed by each algorithm for 10 inputs

SL#	SURF (Seconds)	ICP-BP (Seconds)	EMD (Seconds)
1	2.4961	281.9537	15.1807
2	1.0221	56.6271	11.5964
3	3.4673	943.5975	11.6985
4	2.0008	202.7352	11.6312
5	1.2311	83.5315	11.5829
6	1.1812	84.7508	11.2573
7	1.3165	100.7686	11.0771
8	1.1879	82.3641	11.2365
9	1.2721	86.857	11.2313
10	1.339	132.6327	11.2215
Average	**1.65141**	**205.5818**	**11.77134**

Figure 19. Time elapsed by each algorithm for average of 10 inputs in graphical format



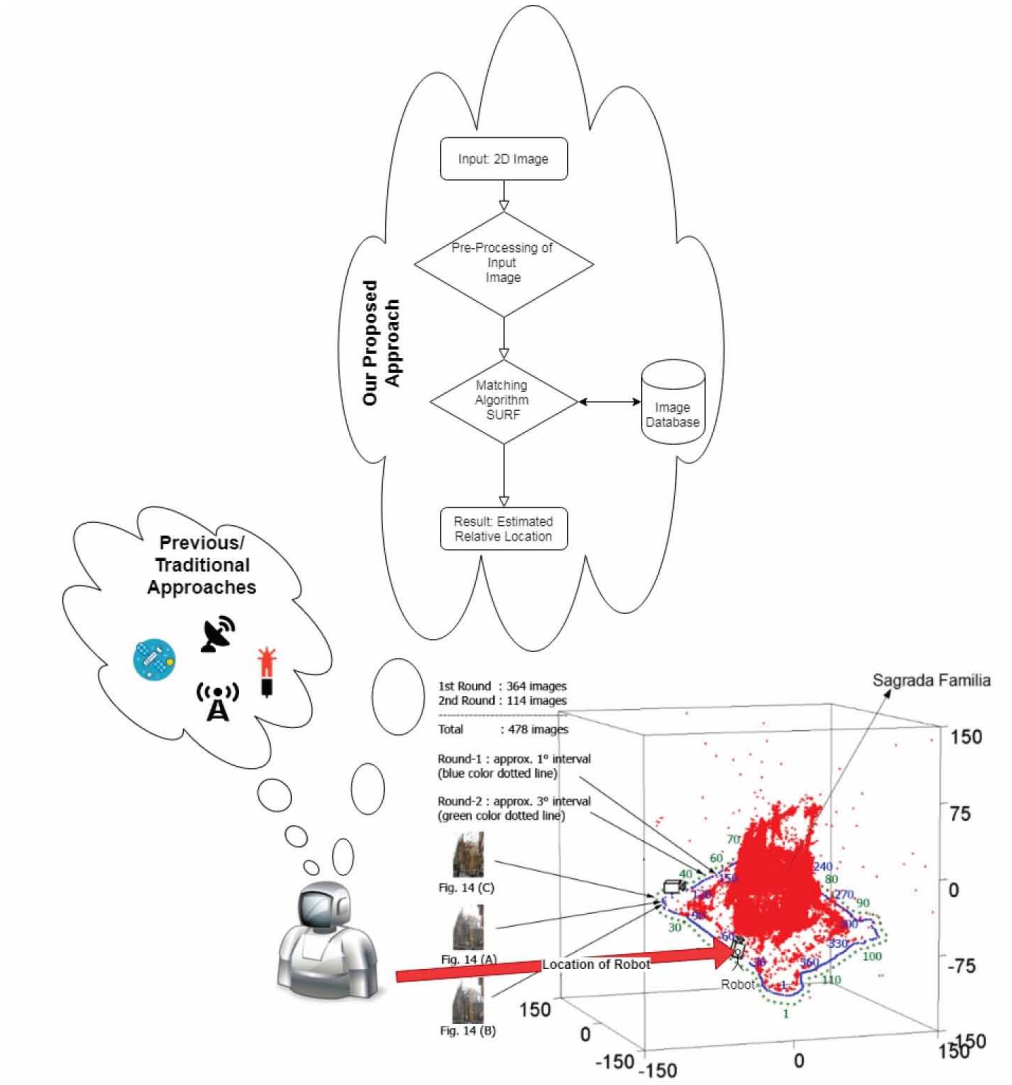
It can be deduced from Table 8 and Figure 19 that SURF is 124.50 and 7.12 times faster than ICP-BP and EMD respectively. It is noteworthy that SURF works with feature extraction values of input and reference images (Table 1, column 9-72) whose length is 64 only. On the other hand, ICP iteratively revises the transformation (combination of translation and rotation) to minimize the distance from the input to the reference image position (considering the salient 3D coordinates) and then BP is used to calculate the distance between input and reference images. Thus, ICP-BP takes

much longer time than SURF to find out the minimum distance. Whereas EMD uses the histogram of input and reference images (resolution is 420 x 720 around) to measure the minimum distance which also longer than feature extraction values and hence takes more time than SURF. So, average elapsed time calculation can be considered as justified where SURF (1.65141s) is better than ICP-BP (205.5818s) and EMD (11.77134s).

Based on the above results and discussion, it can be stated that autonomous robot can move easily using an internal image database without any help of external tools or connectivity. It has also been found that SURF is the best approach to find out an autonomous robots' position using the image database.

The entire flowchart of the method has been given below where it is clearly depicted that our method will add value to other methods if external help will not be available at any time (Figure 20).

Figure 20. Flowchart of the proposed method



CONCLUSION AND FUTURE WORK

An innovative approach has been presented in this paper to find the current location of an autonomous robot in the area where there is no GPS, Internet, or any other network connectivity. In this approach, all the images and 2D, 3D coordinates of any specific area are saved in an internal database. If the robot is placed anywhere (within the area), it will take a new picture and match it with the images stored in the internal database to get its current position. We have presented the implementation of the proposed method using three algorithms SURF, ICP-BP, and EMD. A comparison of these three algorithms has been presented based on accuracy and elapsed time. According to the results and evaluation, SURF is found better than ICP-BP and EMD to get the position of an autonomous robot in offline processing. It is noticeable that a detailed overview of several existing methods using various external helping tools for localization has been discussed in Introduction section. We believe that the proposed method will also add value to the other methods if none of those external tools are available.

In this research work, only known places have been considered for which we have images of 360 degrees. Moreover, the sunlight, sunny, rainy or cloudy days have not been taken into account. In future, we hope to enrich the image database by including more images automatically for both known and unknown places. It is also expected to extend for a larger metropolitan area considering the mobility of other objects. We hope the experiment will be accomplished in various seasons and sun-light variation even in evening and night mode in the next phase.

REFERENCES

Autonomous Robots. (2020). What Are Autonomous Robots. Last accessed at Jan-2020. Retrieved from https://waypointrobotics.com/blog/what-autonomous-robots

Bay, H., Ess, A., Tuytelaars, T., & Gool, L. V. (2008a). SURF: Speeded Up Robust Features. *International Journal of Computer Vision and Image Understanding*, *110*(3), 346–359. doi:10.1016/j.cviu.2007.09.014

Bay, H., Ess, A., Tuytelaars, T., & Gool, L. V. (2008b). SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding*, *110*(3), 346–359. doi:10.1016/j.cviu.2007.09.014

Bedkowski, J., Maslowski, A., & Cubber, G. D. (2012). Real time 3D localization and mapping for USAR robotic application. *Industrial Robot: An International Journal*, *39*(5), 464–474. doi:10.1108/01439911211249751

Benois-Pineau, J., Precioso, F., & Cord, M. (2012). *Visual Indexing and Retrieval*. Springer Science & Business Media; doi:10.1007/978-1-4614-3588-4

Berrabah, S. A., & Bedkowski, J. (2008). *Robot Localization based on geo-referenced Images and Graphic Methods*. Benicassim, Spain: IARP WS RISE.

Besl, P. J., & McKay, N. D. (1992). A Method for Registration of 3-D Shapes. IEEE Trans. on Pattern Analysis and Machine Intelligence, 14 (2), 239–256.

Burgard, W., Fox, D., & Thrun, S. (1997). Active mobile robot localization by entropy minimization. In *Proceedings Second EUROMICRO Workshop on Advanced Mobile Robots*, (pp. 155-162). doi:10.1109/EURBOT.1997.633623

Conversion of JPEG format images into PGM format. (2020). *IRFAN View.* Retrieved from https://www.irfanview.net/

Feature extraction. (2019). *Feature extraction for compact representation of image*. Retrieved from https://www.mathworks.com/discovery/feature-extraction.html

Garrell, A., & Sanfeliu, A. (2012). Cooperative social robots to accompany groups of people. *The International Journal of Robotics Research*, *31*(13), 1675–1701. doi:10.1177/0278364912459278

Ghosh, P., Pandey, A., & Pati, U. C. (2015). Comparison of Different Feature Detection Techniques for Image Mosaicing. *ACCENTS Transactions on Image Processing and Computer Vision*, *1*(1), 2455–4707.

Goel, P., Roumeliotis, S. I., & Sukhatme, G. S. (1999). Robust localization using relative and absolute position estimates. *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*. doi:10.1109/IROS.1999.812832

Han, S. H., Lee, M. H., & Hashimoto, H. (2000). Image-Based Visual Servoing Control of a SCARA Robot. *H. KSME International Journal*, *14*(782), 782–788. doi:10.1007/BF03184464

Haque, M. M., Shohag, Md., Yasin, A., & Anwar, S. (2013). Mobile Ad-Hoc Network Security: An Overview. *IJSRET*, *2*(8), 504–511.

Holdener & Anthony, T. (2011). HTML5 Geolocation. O'Reilly Media.

Huang, W.-T., Tsai, C.-L., & Lin, H.-Y. (2012). Mobile Robot Localization using Ceiling Landmarks and Images Captured from an RGB-D Camera. In *The 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics,* (pp. 855-860). Kaohsiung, Taiwan: IEEE.

Magee, M., & Aggarwal, J. (1995). Robot self-localization using visual reasoning relative to a single target object. *Pattern Recognition*, *28*(2), 125–134. doi:10.1016/0031-3203(94)00091-Y

Matlab function. (n.d.). *Read elapsed time from stopwatch.* Retrieved from https://se.mathworks.com/help/matlab/ref/toc.html

Matlab function. (n.d.). Retrieved from https://www.mathworks.com/help/matlab/ref/tic.html

Riesen, K., Neuhaus, M., & Bunke, H. (2007). Bipartite Graph Matching for Computing the Edit Distance of Graphs. *Proceedings of 6th International Workshop on Graph Based Representations in Pattern Recognition, LNCS 4538*, 1-12. doi:10.1007/978-3-540-72903-7_1

Riesen, & Bunke, H. (2009). Approximate graph edit distance computation by means of bipartite graph matching. *Image Vision Comput., 27*(7), 950-959.

Rubio, A., Villamizar, M., Ferraz, L., Penate-Sanchez, A., Ramisa, A., Simo-Serra, E., & Moreno-Noguer, F. (2015). Efficient monocular pose estimation for complex 3D models. *International Congress on Robotics and Automation*.

Rubner, Y., Tomasi, C., & Guibas, L. J. (1998). A metric for distributions with applications to image databases. *IEEE International Conference on Computer Vision*, (pp. 59-66). doi:10.1109/ICCV.1998.710701

Saurer, O., Baatz, G., Köser, K., Ladický, L., & Pollefeys, M. (2015). Image Based geo-localization in the Alps. *International Journal of Computer Vision*, *116*(3), 1–19.

Serratosa, F., & Cortes, X. (2014). Edit Distance Computed by Fast Bipartite Graph Matching. Academic Press.

Serratosa, F., & Sanfeliu, A. (2006). Signatures versus Histograms: Definitions, Distances and Algorithms. *Pattern Recognition*, *39*(5), 921–934. doi:10.1016/j.patcog.2005.12.005

Shwe, L. L. T., & Win, W. Y. (2017). Vision-Based Mobile Robot Self-localization and Mapping System for Indoor Environment. *American Scientific Research Journal for Engineering, Technology, and Sciences*, *38*(1), 525–530.

Snavely, N., & Todorovic, S. (2011). From contours to 3D object detection and pose estimation. *International Congress on Computer Vision*.

Trinh, L. A., Thang, N. D., Kim, D., Lee, S., & Chang, S. (2012). Application of Matrix Pencil Algorithm to Mobile Robot Localization Using Hybrid DOA/TOA Estimation. *International Journal of Advanced Robotic Systems*, *9*(6), 254–263. doi:10.5772/54712

Visita Virtual. (2020). BASILICA DE LA SAGRADA FAMILIA. Retrieved from https://sagradafamilia.org/visita-virtual

Wu, G., Zheng, J., Bao, J., & Li, S. (2018). Mobile robot location algorithm based on image processing technology. *EURASIP Journal on Image and Video Processing*, *2018*(107), 1–8. doi:10.1186/s13640-018-0352-0

Wu, Y. (2016). *Image based camera localization: an overview.* CoRR, abs/1610.03660

Yun, J., Lyu, E. T., & Lee, J. M. (2006). Image-Based Absolute Positioning System for Mobile Robot Navigation. In D. Y. Yeung, J. T. Kwok, A. Fred, F. Roli, & D. de Ridder (Eds.), *Structural, Syntactic, and Statistical Pattern Recognition* (pp. 261–269). Berlin: Springer. doi:10.1007/11815921_28

Abu Sadat Mohammed Yasin was born in Chattagram, Bangladesh, in 1985. He received the Bachelor of Science (Engineering) in Computer Science and Engineering from Shahjalal University of Science & Technology, Sylhet, Bangladesh in 2009 and the Masters degree in Computer Engineering from Rovira i Virgili University, Tarragona, Spain in 2015. At present, he is an Assistant Systems Analyst (Deputy Director) of the Central Bank of Bangladesh. His research interest includes Machine Learning, Deep Learning, Computer Vision and Image processing.

Dr. Md. Majharul Haque was born in Comilla, Bangladesh, in 1983. He completed Bachelor of C.S.E. from the University of Development Alternative (UODA) and M.S. degree in IT from University of Dhaka, Bangladesh. In 2018, he has achieved PhD Degree from the University of Dhaka in the ground of Natural Language Processing. Previously, he worked as Assistant Professor (Visiting faculty) at UODA, did consultancy job at Online Content Provider Ltd, Sister Concern of Business Automation Ltd. At present, he has been continuing job as Assistant Systems Analyst (Deputy Director) in the Central Bank of Bangladesh. His research interest includes Natural Language Processing, Machine Learning, Data Science and Image processing.

Dr. Md Nasim Adnan is an Assistant Professor in the Department of Computer Science and Engineering, Jashore University of Science and Technology. Nasim received his B.Sc. in Computer Science and Engineering degree from Khulna University, Bangladesh, M.Sc. in Computer Science and Engineering degree from Bangladesh University of Engineering and Technology (BUET), Bangladesh and Ph.D. from the Charles Sturt University, Australia in 2002, 2010 and 2017 respectively. Previously he worked as an Assistant Professor at the Department of Computer Science and Engineering in the University of Liberal Arts Bangladesh (ULAB). He also served as a Systems Analyst in Bangladesh Bank (the Central Bank of Bangladesh). His research interest includes Data Mining, Software Engineering and E-Commerce. He authored and co-authored several papers on Data Mining, Software Engineering and E-Commerce.

Sonia Rahnuma, was born in Rajshahi, Bangladesh, in 1984. She achieved 4 year Bachelor of Science degree in Computer Science and Engineering from Jahangirnagar University, Savar, Bangladesh in 2005 and Master of Science degree in same subject from the same institution in 2006. She completed a project on 'Barcode enabled answer paper evaluation system' as her Master degree project proposal. She worked as a PROGRAMMER in an IT farm named Satcom IT Ltd. from 2008 to 2010. From 2011 to present she is working in Bangladesh Bank, The Central Bank of Bangladesh as a PROGRAMMER. Ms. Rahnuma is a member of Bangladesh Computer Society, a society for ICT professionals since 2006.

Anowar Hossain is a passionate software professional with a spur to accept new challenges. Always eager to learn new mechanisms to make my contribution more meaningful.

Kallol Naha was born in Comilla, Bangladesh, in 1984. He received the B.Sc. degree from the Shahjalal University of Science and Technology, Sylhet, Bangladesh in Computer Science and Engineering in 2009 and received Master degree from Universitat Rovira i Virgili, Tarragona, Spain in Computer Security and Intelligent Systems In 2016. He is now working as a remote researcher and developer in a web based "Field Loadable Software" management tool named FLS-DESK in Germany. Here he has researched and designed intelligent learning algorithm so that aircraft software updates can be done automatically for any PDL (Portable Data Loader)-Pads according to demand. He also developed automatic distribution of FLS to Portable Data Loaders over the air or via Ethernet.

Mohammod Akbar Kabir received his M.Sc. and B.Sc. (Hons) in Computer science from university of Dhaka in 2000 and 1998 Respectively. Currently he is working as an associate professor in the dept. of Economics, University of Dhaka. He also served as an assistant programmer in the dept. of ITOCD, Bangladesh Bank and as a lecturer, Dept. of computer Science, Dhaka City College, Dhaka. Recently he received Master of Economics (Environmental Economics) degree from Dhaka School of Economics, Bangladesh. His research interest includes environmental valuation, sustainable development and information economics.

Francesc Serratosa was born in Barcelona, Catalonia (Spain) in 1967. He received his computer science engineering degree from the Universitat Politècnica de Catalunya (Barcelona) in 1993. He received his Ph.D. from the same University in 2000. He is currently a full professor of computer science at the Universitat Rovira i Virgili at Tarragona, Catalonia (Spain). Since 1993, he has been active in research in the areas of Computer Vision, Robotics, Structural Pattern Recognition, Machine Learning and Biometrics. He has published more than 100 papers and he is an active reviewer in some congresses and journals. He is giving classes on computer vision and biometrics at Universitat Rovira i Virgili and Universitat Oberta de Catalunya. He is the coordinator of the Joint Master of Safety and Technology of Information and communications at Universitat Rovira i Virgili (MISTIC). He has been the coordinator of the PhD course on Computer Science and Security at the Universitat Rovira i Virgili from 2006 to 2012. He has been involved in more than 10 research projects.