FITTING SPHERICAL OBJECTS IN 3D POINT CLOUD USING THE GEOMETRICAL CONSTRAINTS

Le Van Hung\textsuperscript{1,2}, Vu Hai\textsuperscript{1}, Nguyen Thi Thuy\textsuperscript{3}, Le Thi Lan\textsuperscript{1}, Tran Thi Thanh Hai\textsuperscript{1}

Abstract

Estimating parameters of a primitive shape from a 3-D point cloud usually meets difficulties due to noises of data require a huge amount of computational time. The real point cloud of objects in the 3D environment have many noise and may be obscured when they are captured from Kinect version 1. In this paper, we utilize and analyse of spherical estimation steps from the point cloud data in Schnabel et al. [1]. From this, we propose a geometrical constraint to search 'good' samples for estimating spherical model and then apply to a new robust estimator named GCSAC (Geometrical Constraint SAmple Consensus). The proposed GCSAC algorithm takes into account geometrical constraints to construct qualified samples. Instead of randomly drawing minimal subset sample, we observe that explicit geometrical constraint of spherical model could drive sampling procedures. At each iteration of GCSAC, the minimal subset sample is selected by two criteria (1) They ensure consistency with the estimated model via a roughly inlier ratio evaluation; (2) The samples satisfy geometrical constraints of the interested objects. Based on the qualified samples, model estimation and verification procedures of a robust estimator are deployed in GCSAC. Comparing with the common robust estimators of RANSAC family (RANSAC, PROSAC, MLESAC, MSAC, LO-RANSAC and NAPSAC), GCSAC overperforms in term of both precisions of the estimated model and computational time. The implementations and evaluation datasets of the proposed method are made publicly available.

Ước lượng các tham số của một cấu trúc hình học cơ bản (hình cầu, hình trụ, hình nón ...) từ một tập đám mây điểm 3-D thu từ cảm biến Kinect thường gặp các khó khăn do dữ liệu có nhiễu và đối với thiết kế tính toán lơn. Trong bài báo này, chúng tôi đề xuất thuật toán ước lượng đối tượng hình học cơ bản với ý tưởng chính là tìm ra các mẫu "tốt" cho việc ước lượng mô hình hình học cơ bản nhằm giảm thiểu tính toán cũng như tăng độ chính xác của mô hình dự đoán. Thuật toán đề xuất có tên GCSAC (Geometrical Constraints Samples Consensus - Răng buộc hình học và sự đồng thuận của các mẫu) được xây dựng dự trên thuật toán ước lượng mạnh RANSAC (RANdom SAmple Consensus). GCSAC sử dụng các phân tích hình học của Schnabel et al. [1] để xác định bộ tham số cho một hình cầu. Thay vì chọn tập mẫu ngẫu nhiên như trong thuật toán RANSAC, trong GCSAC, chúng tôi đề xuất một approach về hình học để lựa chọn các mẫu "tốt" có đủ điều kiện cho dự đoán mô hình. Sử dụng trên các mẫu có chất lượng, mô hình sẽ được "đủ đoán" và "kiểm chứng" trong GCSAC như đối với các bộ ước lượng RANSAC. Tại mỗi lần lặp lại của GCSAC, tập hợp mẫu nhỏ nhất được lựa chọn theo hai tiêu chí: (1) Đảm bảo tính nhất quán với mô hình ước lượng thông qua một đánh giá tỷ lệ số mẫu inlier; (2) Các mẫu phải thỏa mãn ràng buộc hình học

\textsuperscript{1} International Research Institute MICA, HUST - CNRS/UMI-2954 - GRENOBLE INP, Vietnam, \textsuperscript{2} Tan Trao University, \textsuperscript{3}Vietnam National University of Agriculture
Evaluating parameters of a primitive shape using robust estimators is a fundamental research in the fields of robotic and computer vision. To build the aiding system for visually impaired people or robot control system for object grasping then objects detection, pose estimation of object in the 3D environment are the important steps. Therein, the object detection that using the primitive shapes detection from the point cloud data of objects is a method. Currently, many robust estimators have been proposed such as Hough Transform [2], RANSAC [3], RANSAC variations (e.g., MLESAC [4], PROSAC [5], LO-RANSAC [6], etc). RANSAC is an efficient iteration algorithm that is proposed by Fischler et al. [3] since 1981. Originally, a RANSAC paradigm draws randomly a Minimal Sample Set (MSS) from a point cloud data without any assumptions. As a result, RANSAC must run a relatively large number of iterations to find an optimal solution before stopping criterion. To improve performances, RANSAC-based methods [7] focus on either a better hypothesis from random samples or higher quality of the samples satisfying the estimated model. In this paper, we tackle a new sampling procedure which utilizes geometrical constraints to qualify a MSS. The MSS consisting of good samples is expected to generate a consensus set which quickly reaches to the maximal inlier ratio. Consequently, the number of iterations could be adaptively updated (as the termination manner of the adaptive RANSAC [8]). In GCSAC, we utilize the maximum log-likelihood of MLESAC algorithm to evaluate the estimated model. Finally, the effectiveness of the proposed method is confirmed by fitting results of a spherical object. The evaluations compare the performances of the proposed method and common RANSACs as original RANSAC, PROSAC, MLESAC, MSAC [4], LO-SAC,
NAPSAC [9]. The implementations of the proposed method and evaluation datasets are made publicly available.

2. Related work

For a general introduction and performances of RANSAC family, readers can refer to good surveys in [10], [7]. In the context of this research, we briefly survey related works which are categorized into two topics. First, efficient schemes on the selection of minimal subset of samples for RANSAC-based robust estimators; and second, techniques for estimating a primitive shape, that focus into the fitting a spherical.

For the first topic, because the original RANSAC is very general with a straightforward implementation, it always requires considerable computational time. Many RANSAC variants have been proposed with further optimization for a minimal sample set (MSS) selection. Progressive Sample Consensus or PROSAC [5] orders quality of samples through a similarity function of two corresponding points in the context of finding good matching features between a pair of images. In PROSAC algorithm, the most promising hypotheses are attempted earlier, therefore drawing the samples is implemented in a more meaningful order. However, PROSAC faces critical issues for defining the similarity function. LO-RANSAC [6] and its fixed version LO\textsuperscript{+}-RANSAC [11] add local optimization steps within RANSAC to improve accuracy. To speed up the computation, adaptive RANSAC [8] probes the data via the consensus sets in order to adaptively determine the number of selected samples. The algorithm is immediately terminated when a smaller number of iterations has been obtained. With the proposed method, the good samples are expected to generate the best model as fast as possible. Therefore, the termination condition of the adaptive RANSAC [8] should be explored. Recently, USAC [12] introduces a new frame-work for a robust estimator. In the USAC framework, some strategies such as the sample check (Stage 1b) or the model check (Stage 2b), before and after model estimation, respectively, are similar to our ideas in this work. However, USAC does not really deploy an estimator for primitive shape(s) from a point cloud. A recent work [13] proposes to use geometric verification within a RANSAC framework. The authors deployed several check procedures such as sample relative configuration check based on the epipolar geometry. Rather than the "check" procedures, our strategies anticipate achieving the best model as soon as possible. Therefore, the number of iterations is significantly reduced thanks to the results of the search for good sample process. The RANSAC-based algorithm used in the method of Chen et al. [14] and Aiger et al. [15] for registering of partially overlapping range images and partial surfaces of a 3D object.

For primitive shape estimation from 3-D point clouds, readers can refer to a survey on feature-based techniques [16]. Some fitting techniques, for instance, multiscale superquadric fitting in [17], Hough transform in [18], are commonly used. Marco et al. [19], Anas et al. [20] used the 3D Hough Transform to estimate, extract sphere from point cloud data. However, the robust estimators (e.g., RANSAC family [7]) are always
preferred techniques. Original RANSAC [3] demonstrates itself robust performances in estimating cylinders from range data. The authors in [21] and [1] formulate primitive shapes (e.g., line, plane, cylinder, sphere, cone) using two to seven parameters such as a cylinder has seven parameters, a sphere has four parameters, a cone has seven parameters, etc. Schnabel et al. [1] defines primitive shapes through some samples and their normal vectors. In this study, geometrical analysis of a sphere in [1] is adopted for defining criteria of the qualified samples as well as for estimating parameters of the interested model from a 3-D point cloud.

3. Proposed method

3.1. Overview of the proposed robust estimator - GCSAC

To estimate parameters of a primitive shape, all of RANSAC variations in the RANSAC family [7] (RANSAC, MLESAC, PROSAC, MSAC, LOSAC, NAPSAC, .etc) implemented some steps as follows: drawing randomly a Minimal Sample Set (MSS)(RANSAC, MLESAC, MSAC) or semi-random (PROSAC) or using constraints of the sample’s distribution (NAPSAC); estimating the model; evaluating the model and choosing the best model. This scheme is repeated $K$ iterations. All of them are shown in the top panel of Fig. 1.

In this study, abilities of the proposed robust estimator (GCSAC) will be consolidated with a sphere. An overview of GCSAC algorithms is shown in the button panel of Fig. 1. Following sections will explain the details of GCSAC’s implementations.

The proposed GCSAC constructs a MSS by the random sampling and sampling using geometrical constraints of each primitive shape estimation, the aim of it is the generating the consensus of samples to be easy. To do this, a low inlier threshold is predetermined. After only (few) random sampling iterations, the candidates of good samples could be achieved. Once initial MSS is established, its samples will be updated by the qualified one (or good sample) so that the geometrical constraints of the interested object is satisfied. The estimated model is evaluated according to Maximum Log-likelihood criteria as MLESAC [4]. The final step is to determine the termination condition, which is adopted from the adaptive RANSAC algorithm [8]. Once the higher inlier ratio is obtained, the criterion termination $K$ for determining the number of sample selection is updated by:

$$K = \frac{\log(1 - p)}{\log(1 - w^m)}$$

(1)

where $p$ is the probability to find a model describing the data, $m$ is the minimal number of samples to estimate a model, $w$ is percentage of inliers in the point cloud. While $p$ often to set a fixed value (e.g., $p = 0.99$ as a conservative probability), $K$ therefore depends on $w$ and $m$. The algorithm terminates as soon as the number of iterations of current estimation is less than that has already been performed.
Details of the GCSAC’s implementation are given in Fig. 1. Obviously, the criteria which define the good samples are the most important. Based on the idea of adaptive RANSAC [8] to probe initial samples, GCSAC starts from roughly select of initial good samples. At each iteration, we assume that the worst case of inlier ratio \( w_t = 0.1(10\%) \) is determined, to initialize the stack \( U_n^* \), where \( U_n^* \) is used to store \( m - 1 \) kept points and its inlier ratio \( w_i \). A consensus set, therefore containing more than 10\% of the data is easily found. A model is estimated from \( m \) random samples. As estimating a sphere is \( m = 2 \) [1]. After that, \( U_n^* \) is reset. \( m \) samples and the inlier ratio \( w_i \) of the estimated model is stored into \( U_n^* \) if \( w_i \) is equal to or greater than \( w_t \). And then, the MSS utilizes \( m - 1 \) kept good samples. The remaining \( m^{th} \) sample will be replaced.
by a better one which best satisfies the geometrical constraints of the interested shape. The good samples which satisfy the geometrical principles of a primitive shape are explained in Section 3-B. If none of the iterations find out that satisfies \( w_i \geq w_t \), the estimation algorithm degrades to the original MLESAC. The inlier ratio of the iteration depends on the threshold \( T \), which chooses an optimal \( T \) value is out of the scope of this research.

3.2. Geometrical analyses and constraints for qualifying good samples

The geometrical model of a spherical object can be presented by four geometrical parameters [1]. In following sections, the principles of 3-D sphere are explained. Based on this geometrical analysis, the related constraints are given to select good samples.

3.2.1. Geometrical analysis for a spherical object: A sphere is determined by the following parameters: a centroid point which is denoted as \( I_{sp}(x_0, y_0, z_0) \); its radius \( R_{sp} \).

To estimate sphere’s parameters, Schnabel et al. [1] propose to use two points \((p_1, p_2)\) with their corresponding normal vectors \((n_1, n_2)\) (see Fig. 2(a)). The centroid \( I_{sp} \) (a pink point Fig. 2(c)) is a middle point of the shortest line (a green line of Fig. 2(b)) which segments two lines given by \((p_1, n_1)\) and \((p_2, n_2)\). Identifying two lines \( pa = p_1 + t \cdot n_1 \) and \( pb = p_2 + t \cdot n_2 \) are shown in Fig. 2(b). \( t \) is the parameter of two lines \( pa, pb \) formulations. The radius \( R_{sp} \) is determined by averaging the distance of \( I_{sp} \) to \( p_1 \) and \( I_{sp} \) to \( p_2 \). The estimated sphere is shown in Fig. 2(d).

The proposed geometrical constraints for fitting spherical objects: As above
denoted, a sphere is estimated from two points \((p_1, p_2)\) and their normal vectors \((n_1, n_2)\). In GCSAC, once stack \(U^*_n\) consisting of initial good samples is specified, we store \(p_1\) and search \(p_2\) in \(U_n\) and difference \(p_1 \in \{U_n \setminus p_1\}\). We observe that to generate a sphere, the triangle \((p_1Isp\ p_2)\) should be an isosceles, as shown in Fig. 2(e). This observation could be formulated by:

\[
sh_p = \arg\min_{p_2 \in \{U_n \setminus p_1\}} \left\{ ||p_1Isp|| - ||p_2Isp|| \right\}
\]  

(2)

The geometrical constraints in Eq. (2) means that if \(sh_p\) is close to 0 then the triangle \(p_1Isp\ p_2\) is nearly isosceles one.

4. Experimental result

4.1. Dataset

The proposed method is warped by C++ programs using a PCL 1.7 library on a PC with Core i5 processor and 8G RAM. The program runs sequentially as a single thread. We have evaluated performances of GCSAC on synthesized datasets of the cylinder, sphere and cone. Each dataset consists of six different subsets. Characteristics of each subset are described in Table 1. Major differences could be \(\sigma\) of the normal distribution for generating outlier/inlier data; or the spatial distribution of inliers.

For the sphere dataset (‘first sphere’), they are denoted from \(dC_1\) to \(dC_6\). In each subset \(dC_i\), inlier ratio is increased by a step of 5% from 15 to 80%. Therefore, there are fourteen point clouds. They are denoted \(dS_1\) to \(dS_{14}\). They have the maximal distance to true sphere 0.025 (or 2.5cm). Outliers are generated randomly in the limitation as Tab. 1. A point cloud \(dS_i\) consists of 3000 points. These point clouds are generated from the curved surface of a true sphere: \(x^2+y^2+z^2 = 1\). We also generate random points outside the surface of the sphere as outliers.

Moreover, we also evaluated on a real datasets. This is the dataset of two balls in four scenes, each scene has been included 500 frames and each frame has a ball on the table. It named ‘second sphere’. The setup of our experiment implemented a similar as [22] and is illustrated in Fig. 3.

To separate the data of a ball on the table, we have to implement some steps as below. First is the table plane detection that used the method in [23]. After that, the original coordinate is rotated and translated that the y-axis is parallel with the normal vector of the table plane as Fig. 3. The point cloud data of balls is separated with the point cloud data of the table plane. It is illustrated in Fig. 5.

4.2. Evaluation Measurements

Let notate a ground-truth of model \(M_t(x_t, y_t, z_t, r_t)\) and the estimated one \(M_e(x_e, y_e, z_e, r_e)\) where \((x_t, y_t, z_t)\), \((x_e, y_e, z_e)\) are the coordinates of the center points,
Fig. 3. Illustrating of our setup to collect the dataset

Fig. 4. Point clouds of \( dC_1, dC_2, dC_3 \) of the ’second sphere’ dataset (the synthesized data) in case of 50% inlier ratio. The red points are inliers, whereas blue points are outliers.

Fig. 5. Illustrating the separating the point cloud data of a ball in the scene.

\( r_t, r_e \) are the radius. To evaluate the performance of the proposed method, we use following measurements:

- Let denote the relative error \( E_{\omega} \) of the estimated inlier ratio. The smaller \( E_{\omega} \) is,
Table 1. The characteristics of the generated sphere dataset (synthesized dataset)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Characteristics of the generalized data</th>
<th>Spread of outliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>dC1, dC4</td>
<td>1 Around of a sphere [-3, 3], [-4, 4]</td>
<td></td>
</tr>
<tr>
<td>dC2, dC5</td>
<td>1 Around of a sphere [-3, 3], [-4, 4]</td>
<td></td>
</tr>
<tr>
<td>dC3, dC6</td>
<td>1 one half of a sphere [-3, 3], [-4, 4]</td>
<td></td>
</tr>
</tbody>
</table>

the better the algorithm is.

\[ E_w = \frac{|w - w_{gt}|}{w_{gt}} \times 100 \]  

(3)

where \( w_{gt} \) is the defined inlier ratio of ground-truth; \( w \) is the inlier ratio of the estimated model.

\[ w = \frac{\# \text{inliers}}{\# \text{number of samples}} \]  

(4)

- The total distance errors \( S_d \) [24] is calculated by summation of distances from any point \( p_j \) to the estimated model \( M_e \). \( S_d \) is defined by:

\[ S_d = \sum_{j=1}^{N} d(p_j, M_e) \]  

(5)

- The processing time \( t_p \) is measured in milliseconds (ms). The smaller \( t_p \) is the faster the algorithm is.

- The relative error of the estimated center (only for synthesized datasets) \( E_d \) is Euclidean distance of the estimated center \( E_e \) and the truth one \( E_t \). It is defined by:

\[ E_d = |E_e - E_t| \]  

(6)

- The relative error of the estimated radius \( E_r \) is different of the estimated radius \( r_e \) and the truth one \( r_t \). It is defined by:

\[ E_r = \frac{|r_e - r_t|}{r_t} \times 100\% \]  

(7)

The proposed method (GCSAC) is compared with six common ones in RANSAC family. They are original RANSAC, PROSAC, MLESAC, MSAC, NAPSAC, LO-RANSAC. For setting the parameters, we fixed thresholds of the estimators with \( T = 0.05 \) (or 5cm), \( w_t = 0.1 \), \( s_r = 3 \) (cm). \( T \) is a distance threshold to set a data point to be inlier or outlier. \( s_r \) is the radius of a sphere when using NAPSAC algorithm. For the fair evaluations, \( T \) is set equally for all seven fitting methods.
4.3. The evaluation results

The performances of each method of the synthesized datasets are reported in Tab. 2. For whole three datasets, GCSAC obtains the highest accuracy and lowest computational time. More generally, even using same criteria as MLESAC, the proposed GCSAC obtains better estimated model as shown by $E_w$ measurements. The experimental results also confirmed the proposed constraints are working well with different primitive shapes. Although $E_w$ of the sphere dataset is high ($E_w = 19.44\%$), this result is still better than the result of the compared methods. Among the compared RANSACs, it is interesting that original RANSAC generally give stable results for estimating a sphere. However, original RANSAC requires a high computational time. The proposed GSSAC estimates the models slightly better than the original RANSAC.

![Fig. 6. Illustrating the fitting sphere of GCSAC and some RANSAC variations on the synthesized datasets, which have 15% inlier ratio. Red points are inlier points, blue points are outlier points. The estimated sphere is a green sphere.](image)

Table 2. The average evaluation results of synthesized datasets. The synthesized datasets were repeated 50 times for statistically representative results.

<table>
<thead>
<tr>
<th>Dataset/Method</th>
<th>Measure</th>
<th>RANSAC</th>
<th>PROSAC</th>
<th>MLESAC</th>
<th>MSAC</th>
<th>LOSAC</th>
<th>NAPSAC</th>
<th>GCSAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘first sphere’</td>
<td>$E_w(%)$</td>
<td>23.01</td>
<td>31.53</td>
<td>85.65</td>
<td>33.43</td>
<td>23.63</td>
<td>57.76</td>
<td>19.44</td>
</tr>
<tr>
<td></td>
<td>$S_d$</td>
<td>3801.95</td>
<td>3803.62</td>
<td>3774.77</td>
<td>3804.27</td>
<td>3558.06</td>
<td>3904.22</td>
<td>3452.88</td>
</tr>
<tr>
<td></td>
<td>$t_p(ms)$</td>
<td>10.68</td>
<td>23.45</td>
<td>1728.21</td>
<td>9.46</td>
<td>31.57</td>
<td>2.96</td>
<td>6.48</td>
</tr>
<tr>
<td></td>
<td>$E_d(cm)$</td>
<td>0.05</td>
<td>0.07</td>
<td>1.71</td>
<td>0.08</td>
<td>0.21</td>
<td>0.97</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>$E_r(%)$</td>
<td>2.92</td>
<td>4.12</td>
<td>203.60</td>
<td>5.15</td>
<td>17.32</td>
<td>63.60</td>
<td>2.61</td>
</tr>
</tbody>
</table>

Table 3 also shown the fitting results of GCSAC method are more accurate than RANSAC variations. The results on the ‘second sphere’ are high on all of the methods
Table 3. The average evaluation results on the 'second sphere' datasets. The real datasets were repeated 20 times for statistically representative results.

<table>
<thead>
<tr>
<th>Dataset/Method</th>
<th>Measure</th>
<th>RANSAC</th>
<th>PROSAC</th>
<th>MLESAC</th>
<th>MSAC</th>
<th>LOSAC</th>
<th>NAPSAC</th>
<th>GCSAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>'second sphere'</td>
<td>( w(%) )</td>
<td>99.77</td>
<td>99.98</td>
<td>99.83</td>
<td>99.80</td>
<td>99.78</td>
<td>98.20</td>
<td>100.00</td>
</tr>
<tr>
<td></td>
<td>( S_d )</td>
<td>29.60</td>
<td>26.62</td>
<td>29.38</td>
<td>29.37</td>
<td>28.77</td>
<td>35.55</td>
<td>11.31</td>
</tr>
<tr>
<td></td>
<td>( t_p \text{(ms)} )</td>
<td>3.44</td>
<td>3.43</td>
<td>4.17</td>
<td>2.97</td>
<td>7.82</td>
<td>4.11</td>
<td>2.93</td>
</tr>
<tr>
<td></td>
<td>( E_r \text{(%)} )</td>
<td>30.56</td>
<td>26.55</td>
<td>30.36</td>
<td>30.38</td>
<td>31.05</td>
<td>35.72</td>
<td>14.08</td>
</tr>
</tbody>
</table>

**Fig. 7.** Results of ball and cone fittings. The point cloud data of a ball is red points; the estimated spherical is marked as green points.

such as the result of GCSAC when using the \( w \) measure is 100\%, it is illustrated in Fig. 7(a). Due to the ball data has a small noise ratio and the threshold \( T \) to determine inlier points is large (0.05(5cm)). While the radius of a ball is 5cm to 7cm. All of the results shown, the performance of GCSAC is better than the RANSAC variations when implements estimating primitive shapes on the point cloud data, that has low inlier ratio (less than 50\%). They were also shown, can use to the GCSAC algorithm for detecting, finding spherical objects in the real scenario. As a visually impaired person come to a sharing room or a kitchen to find a ball on the floor.

## 5. Conclusions

In this paper, we proposed GCSAC that is a new RANSAC-based robust estimator for fitting the primitive shapes from point clouds. The key idea of GCSAC is the combination of ensuring consistency with the estimated model via a roughly inlier ratio evaluation and geometrical constraints of the interested shapes that help to select good samples for estimating a model. Our proposed method is examined with common shapes (e.g., a sphere). The experimental results confirmed on the synthesized, real datasets that it works well even the point clouds with low inlier ratio. The results of the GCSAC algorithm compared to the RANSAC variations in the RANSAC family. In the future, we
continue to validate GCSAC on other geometrical structures and evaluate the proposed method with the real scenario of detecting multiple objects.

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2017.315

References


Le Van Hung received his M.Sc. at Faculty Information Technology- Hanoi National University of Education (2013). He is now an PhD at International Research Institute MICA HUST-CNRS/UMI - 2954 - INP Grenoble. His research interests include Computer vision, RANSAC and RANSAC variation and 3D object detection, recognition.

Vu Hai received B.E. degree in Electronic and Telecommunications in 1999 and M.E. in Information Processing and Communication in 2002, both from Hanoi University of Science and Technology (HUST). He received Ph.D. in Computer Science from Osaka University, 2009. He has been a lecturer at MICA International Research Institute since 2012. He is interested in computer vision techniques for medical imaging, HCI, robotics.

Nguyen Thi Thuy received her BSc. degree in Math & Informatics in 1994, MSc. in Information Technology from HUST in 2002, and PhD. in Computer science from Graz University of Technology, Austria in 2009. She has been a lecturer at Faculty of Information Technology, Vietnam National University of Agriculture since 1998, head of department of Computer Science since 2011. Her research interest includes object recognition, visual learning, video understanding, statistical methods for computer vision and machine learning.

Le Thi Lan graduated in Information Technology HUST, Vietnam. She obtained MS degree in Signal Processing and Communication from HUST, Vietnam. In 2009, she received her Ph.D. degree at INRIA Sophia Antipolis, France in video retrieval. She is currently lecturer/researcher at Computer Vision Department, HUST, Vietnam. Her research interests include computer vision, content-based indexing and retrieval, video understanding and human-robot interaction.

Tran Thi Thanh Hai graduated in Information Technology from Hanoi University of Science and Technology in 2001. She received ME. then Ph.D In Imagery Vision Robotic from INPG in 2002 and 2006 respectively. Since 2009, she is lecturer/researcher at Computer Vision group, International Institute MICA, Hanoi University of Science and Technology. Her main research interests are visual object recognition, video understanding, human-robot interaction and text detection for applications in Computer Vision.