Robust Multithreaded Object Tracker through Occlusions for Spatial Augmented Reality

Ahyun Lee and Insung Jang

A spatial augmented reality (SAR) system enables a virtual image to be projected onto the surface of a real-world object and the user to intuitively control the image using a tangible interface. However, occlusions frequently occur, such as a sudden change in the lighting environment or the generation of obstacles. We propose a robust object tracker based on a multithreaded system, which can track an object robustly through occlusions. Our multithreaded tracker is divided into two threads: the detection thread detects distinctive features in a frame-to-frame manner, and the tracking thread tracks features periodically using an optical-flow-based tracking method. Consequently, although the speed of the detection thread is considerably slow, we achieve real-time performance owing to the multithreaded configuration. Moreover, the proposed outlier filtering automatically updates a random sample consensus distance threshold for eliminating outliers according to environmental changes. Experimental results show that our approach tracks an object robustly in real-time in an SAR environment where there are frequent occlusions occurring from augmented projection images.

Keywords: Augmented reality, RANSAC, Spatial augmented reality, Stitching, Tracking.

I. Introduction

A spatial augmented reality (SAR) technique is a projector-based form of augmented reality (AR). Using such techniques, virtual images can be directly displayed on the surface of a real-world object, as shown in Fig. 1. Mirage Table [1] by Microsoft is an interactive table based on a projected AR technique. The camera recognizes and detects real-world objects, and the projector projects virtual information onto a table. A future robotic computer (FRC) [2] enables the movement of projector and camera poses using robotic actuators. This means that an FRC can move projection images according to the requirements of the application, such as having a wide projection in a restrictive environment or changing the direction of projection. When a projector and camera are moved, the calculation of their pose is essential for rendering a projection image. This system uses robotic kinematics based on a computer-aided design or recognizes a marker for pose estimation.

Another feature of the SAR technique is that a user can use tangible objects as an interface device. Figure 1(c and d) show examples of SAR systems using tangible objects. Lego Oasis [3] recognizes user-created Lego blocks and projects animated projection images. A curve design game [4] uses yellow paper markers that can be easily produced for designing a curve. These systems provide the user with intuitive control using a tangible interface by projecting a virtual image onto the surface of a real-world object or onto a workplace. They can provide an easy and simple user interface environment for children and the elderly. However, SAR systems are unfavorable for detecting or tracking real-world objects owing to occlusions.

Occlusions refer to sudden changes in the lighting environment or the generation of obstacles between the camera and the tracked object. They are a serious issue in

Manuscript received Aug. 7, 2017; revised Nov. 13, 2017; accepted Nov. 23, 2017.

Ahyun Lee (corresponding author, ahyun@etri.re.kr) and Insung Jang (e4dol2@etri.re.kr) are with the Hyper-connected Communication Research Laboratory, ETRI, Daejeon, Rep. of Korea.

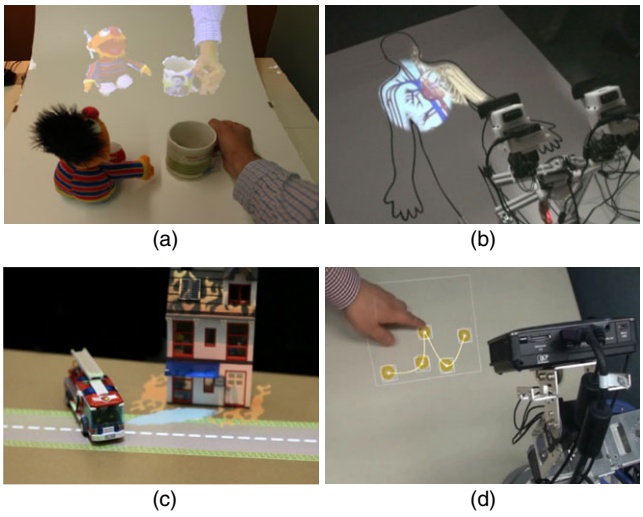


Fig. 1. Examples of SAR systems: (a) MirageTable [1], (b) a future robotic computer [2], (c) Lego Oasis [3], and (d) a curve design game [4].

object tracking when calculating a camera pose in an SAR environment. There are well-known solutions to occlusions, such as a scale-invariant feature transform (SIFT) [5], a speeded-up robust feature (SURF) [6], oriented features from accelerated segment test, and rotated BRIEF (ORB) [7], which are rotation-invariant and resistant to noise. However, these tracking methods incur significant computational costs for achieving real-time performance.

Hybrid feature tracking (HFT) [8] with an SIFT has reliable tracking performance in a real-time system. Although a multithreaded system detects the SIFT features in another thread, it achieves real-time performance for calculating a camera position in the main thread. However, its tracking reliability is low in an SAR environment where there are frequent occlusions from augmented projection images.

Simultaneous localization and mapping (SLAM) is well-known for estimating the camera trajectory while reconstructing the environment [9]. It detects a large number of distinctive features in the environment and uses bundle adjustment to increase the accuracy of camera localizations. However, we use a single camera mounted on a robot arm. Camera localizations are estimated by data-driven robot kinematic control [10]. We only use the proposed method for tracking a movable object. Therefore, the number of features from an object is smaller than that of SLAM approaches. This means that object features can be easily affected by frequent occlusions in the projection-based AR system. Similarly, Wang [11] uses SLAM for camera localizations and tracks a movable object using an RGB-D camera.

SLAM methods use a random sample consensus (RANSAC) [12] to eliminate the errors from detected and matched features [9, 13]. The RANSAC distance threshold must be predetermined by the user. In addition, these papers do not contain the contents of the RANSAC parameters. Cheng [14] presented an automatic way to determine the RANSAC threshold based on the projection error of the RANSAC. We proposed a tracking method that determines a case-specific RANSAC distance threshold based on an optical flow tracker [15]. It enables the threshold to be determined according to the object shape and a scale. However, the threshold also must be predetermined manually by the user.

In this paper, we describe a multithreaded object tracker with outlier filtering, as shown in Fig. 2. It enables the RANSAC distance threshold to be determined according to environmental changes. In addition, the RANSAC threshold can be updated automatically at user-defined intervals in real-time. The rest of this paper is structured as follows. In Section II, we describe an SAR system model and how to render a projection image based on the pose of the projector. In Section III, we describe the proposed multithreaded object tracker with outlier filtering in detail. In Section IV, we evaluate the performance of the proposed tracker and show our experiment results. Finally, in Section V, we provide some concluding remarks.

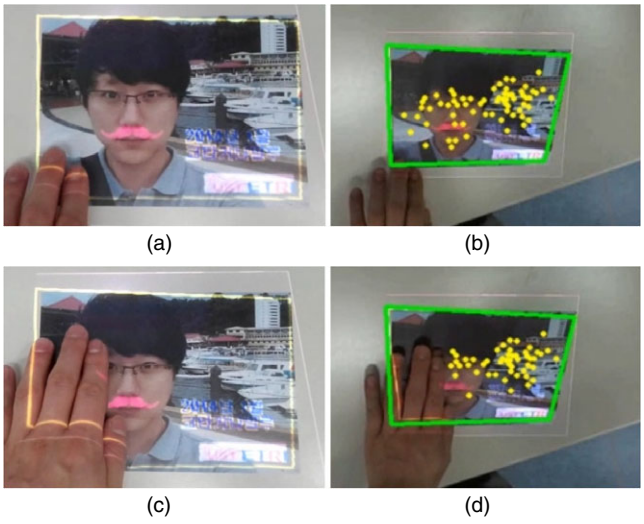


Fig. 2. Tracking example of the proposed object tracker. The yellow points indicate the tracked features in the camera coordinates: (a) augmented projection images on the image, (b) tracking result of (a), (c) occurrence of occlusions by a hand, and (d) errors removed through outlier filtering.

II. SAR System Model

The SAR system implemented in this study was tested using an FRC developed by ETRI [16]. The FRC consists of five actuators and two projector–camera unit pairs. The camera recognizes real-world objects, and the projector projects additional virtual information onto the surfaces of real-world objects. The camera and projector are physically fixed to each other. Therefore, if a projector–camera unit is calibrated in advance, the projector pose can be calculated by simply estimating the camera's extrinsic parameters. In this section, we describe the SAR system model and the calibration method for a projector–camera unit.

1. Projector–Camera Model

The SAR coordinate system is based on a pinhole camera model and must be defined based on the geometric relationship of the projector and through camera calibration [16], [17]. Each transform homography matrix consists of an intrinsic parameter and an extrinsic parameter. The intrinsic parameter $\mathbf{M}$ is a 3 × 3 matrix. Here, α and β are the scale factors of the u and v axes of the image, u_0 and v_0 are the principal points of the coordinate system, and γ is the skew parameter.

$$\mathbf{M} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

$$\mathbf{X}_{wc} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}. \tag{2}$$

The extrinsic parameter $\mathbf{X}_{wc}$ is a 3 × 4 matrix. It is composed of the translation vector $\mathbf{t}$ and rotation matrix $\mathbf{R}$ of the x, y, and z axes. In our system, the camera is calibrated using Zhang's method [18], [19]. Here, $\mathbf{p}_c$ is a point in the calibrated camera coordinates, and $\mathbf{p}_w$ is the corresponding point in the world coordinates, as shown in Fig. 3. In this case, the relationship between them is defined as the camera homography matrix $\mathbf{H}_{wc}$, in which the intrinsic camera parameter $\mathbf{M}_c$ and extrinsic camera parameter $\mathbf{X}_{wc}$ are combined.

$$\mathbf{p}_c = \mathbf{M}_c\mathbf{X}_{wc}\mathbf{p}_w = \mathbf{H}_{wc}\mathbf{p}_w. \tag{3}$$

In the case of projector calibration, we cannot use Zhang's method, which requires knowing the image resolution in advance. It is difficult to know the resolution of the projected image from a projector in advance owing to the offset lens of the projector [16]. Thus, the projector in our system is calibrated using Tsai's method [16], [20]. Here, $\mathbf{p}_p$ is a point in the calibrated projector coordinates, and $\mathbf{p}_w$ is the corresponding point in the world coordinates, as shown in Fig. 3. In this case, the relationship between them is defined as the projector homography matrix $\mathbf{H}_{wp}$, in which the intrinsic projector parameter $\mathbf{M}_p$ and extrinsic projector parameter $\mathbf{X}_{wp}$ are combined.

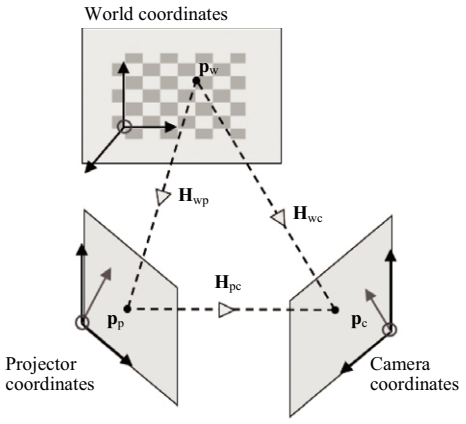


Fig. 3. SAR coordinate system.

$$\mathbf{p}_p = \mathbf{M}_p\mathbf{X}_{wp}\mathbf{p}_w = \mathbf{H}_{wp}\mathbf{p}_w. \tag{4}$$

The relationship matrix $\mathbf{X}_{cp}$ between the projector and the camera is calculated using the inverse matrix of the extrinsic camera parameter and extrinsic projector parameter.

$$\mathbf{X}_{cp} = \mathbf{X}_{wp}\mathbf{X}_{wc}^{-1} = \mathbf{X}_{wp}\mathbf{X}_{cw} = \mathbf{X}_{pc}^{-1}. \tag{5}$$

The projector homography matrix $\mathbf{H}_{wp}$ is calculated using the extrinsic camera parameter, precomputed intrinsic projector parameter, and relationship matrix $\mathbf{X}_{cp}$ between the projector and the camera, which is calculated using (5).

$$\mathbf{M}_p\mathbf{X}_{cp}\mathbf{X}_{wc} = \mathbf{M}_p\mathbf{X}_{wp} = \mathbf{H}_{wp} = \mathbf{H}_{pw}^{-1}. \tag{6}$$

If $\mathbf{p}_c$ is a point in the camera coordinates, and is $\mathbf{p}_p$ is the corresponding point in the projector coordinates, the relationship between them is defined as the matrix $\mathbf{H}_{pc}$. In the real-time process, $\mathbf{H}_{pc}$ is calculated using the camera homography and the inverse matrix of the projector homography:

$$\mathbf{p}_c = \mathbf{H}_{wc}\mathbf{H}_{wp}^{-1}\mathbf{p}_p = \mathbf{H}_{cw}^{-1}\mathbf{H}_{pw}\mathbf{p}_p = \mathbf{H}_{pc}\mathbf{p}_p. \tag{7}$$

In our implemented system, the camera and projector are physically fixed to each other. Therefore, the relation

https://doi.org/10.4218/etrij.2017-0047

matrix $\mathbf{H}_{cp}$ between the projector and the camera is a fixed value in an SAR system.

2. Rendering a Projection Image

In our SAR system, the camera detects a real-world object, and the projector projects a virtual image onto the surface of the object. The projector–camera unit is calibrated based on the projector–camera model described in the previous section. First, we need to calculate the extrinsic camera parameter $\mathbf{X}_{wc}$ to generate a virtual projection image. The perspective-n-point (PnP) estimation using RANSAC between the $\mathbf{p}_w$ features detected in the world coordinates from the object image and the $\mathbf{p}_c$ features detected in the camera coordinates is applied to calculate $\mathbf{X}_{wc}$. Features in the world coordinates must be three-dimensional points for PnP estimation. All z-axis values of $\mathbf{p}_w$ are zero owing to the use of a planar object as the real-world object. The central origin point $(0, 0)$ in the object image is the central origin point $(0, 0, 0)$ of the world coordinates. Finally, the projection image $\mathbf{I}_w$ in the world coordinates is transformed into the projection image $\mathbf{I}_p$ in the projector coordinates.

$$\mathbf{I}_p = \mathbf{M}_p\mathbf{X}_{cp}\mathbf{X}_{wc}\mathbf{I}_w = \mathbf{H}_{wp}\mathbf{I}_w. \quad (8)$$

III. Multithreaded Tracking System

The pose estimation of a projector–camera unit is the most essential part of an SAR system. It is carried out as PnP estimation applied between the scene features detected in an object of the camera input image and the object features detected in the original object of the world coordinates. The object features are detected using an SIFT. The scene features are detected and tracked by the proposed multithreaded tracker during the real-time process.

Our multithreaded tracker is implemented based on HFT [8] and is divided into two threads: a detection thread for detecting distinctive features and a tracking thread for tracking features using a pyramidal Lucas–Kanade tracker (LKT), which is an optical-flow-based tracker [21], [22]. Figure 4 shows the task flow of the multithreaded object tracker system. The tracking thread is applied frame-to-frame for tracking the detected features, whereas the detection thread is applied periodically for detecting newly added SIFT features as the scene features in a camera input image. Owing to occlusions, tracked features are easily removed or deformed through the optical-flow-based

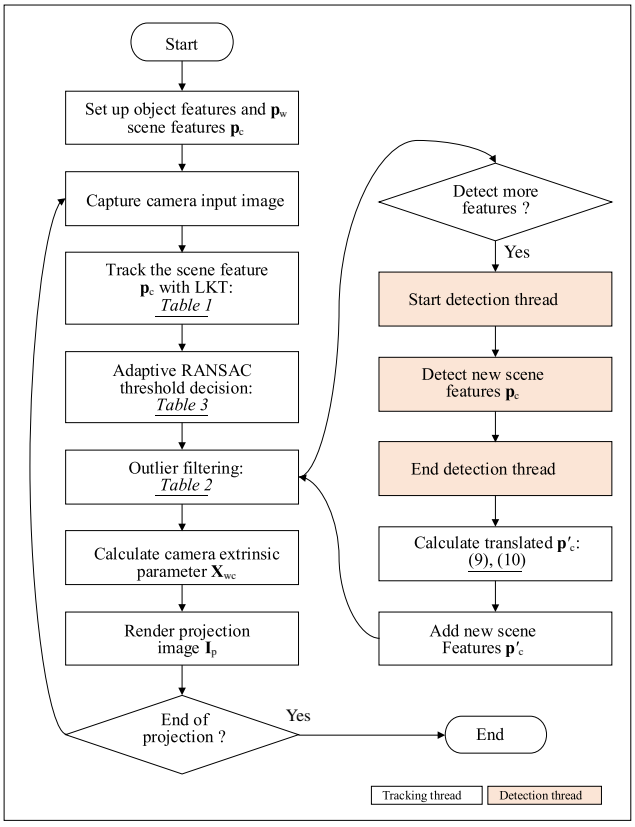


Fig. 4. Work flow of an SAR system using the proposed multithreaded object tracker.

LKT. Therefore, the proposed outlier filtering is imperative for removing errors for robust camera pose estimation.

1. Tracking Thread

The tracking thread is the main thread in an SAR system. Therefore, the real-time tracking thread runs at the same interval as the camera's frame rate, which is indicated with the white blocks shown in Fig. 4. Table 1 summarizes the pseudocode for the tracking thread. When the system starts, we set the object that will be detected and tracked in the loop process. We detect SIFT features as the object features from the original object in the world coordinates. Because the original image is a two-dimensional planar object, all z-axis values of the detected object features are set to zero.

After the loop process in Table 1 starts, the tracking thread tracks the detected scene features in a camera input image. According to the tracking results, the extrinsic camera parameter $\mathbf{X}_{wc}$ is calculated through PnP estimation using RANSAC. The extrinsic projector parameter that is physically fixed to the camera can be

Table 1. Pseudocode of the tracking thread.

```
FUNCTION: MAIN()
  Detect object feature p_w in an object image by an SIFT.
  Detect scene feature p_c in a camera input image
  Set p_c to LKT features p_{c_0}
  i, k = 0
  BEGINLOOP:
    Capture a camera input image.
    Track p_{c_i} by an LKT; its tracked result is p_{c_{i+1}}.
    Adaptive RANSAC threshold decision process.
    k = k + 1
    IF: k = 100,
      Update RANSAC distance threshold.
      k = 0.
    IF: Detection thread is over,
      Calculate transformed pose of detected features.
      Add newly detected scene features to p_{c_{i+1}}.
    Outlier filtering with p_w and p_{c_{i+1}}; its result is p_{c_{i+1}}.
    IF: number of p_{c_{i+1}} is smaller than the user-defined number
      OR user-defined interval,
      Detection thread starts.
    PnP estimation with p_w and p_{c_{i+1}}
    Rendering a projection image.
    i = i + 1
  ENDLOOP
```

estimated using $\mathbf{X}_{wc}$ and $\mathbf{X}_{cp}$ with (6). Here, $\mathbf{X}_{cp}$ is precomputed during the offline process. Finally, a projection image projected by the projector is visualized. Our SAR system includes outlier filtering for robust tracking through environmental changes such as the distance between the camera and the object, the object shape, or occlusions. The proposed outlier filtering uses a case-specific threshold that is automatically updated using the proposed adaptive RANSAC threshold decision.

A. Outlier Filtering

The proposed outlier filtering is applied in a frame-by-frame manner in the tracking thread before PnP estimation. It chooses good features as inliers for tracking with an LKT and eliminates unstable features as outliers that are affected by occlusions. Our approach classifies them into inliers and outliers based on the relationship between the object image and the scene image, as shown in Fig. 5(a and b). The object image is the original image of the real-world object, and the origin of the object image is set as the origin of the world coordinates in Fig. 5(a). The scene image is captured in the loop process shown in Fig. 5(b).

Table 2 summarizes the detailed steps of the proposed outlier filtering. If a transformed feature $\mathbf{p}'_c$ in the scene image is placed out of the case-specific threshold range, it is regarded as an outlier, as shown in Fig. 5(c). Therefore, this implies that the distance threshold is the primary factor for classifying inliers and outliers. We describe the proposed adaptive RANSAC threshold decision in the next section.

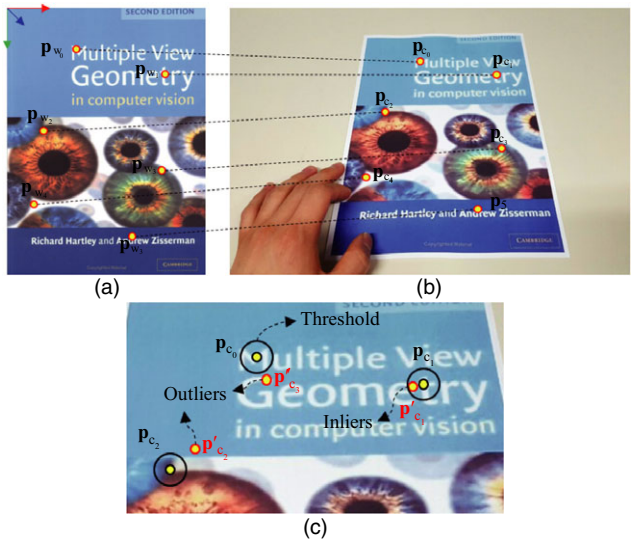


Fig. 5. Detected and tracked features in the object and scene image and the definition of outliers: (a) detected features through an SIFT in the object image, (b) results tracked by an LKT from the scene image in the loop process, and (c) an outlier feature if a transformed feature $\mathbf{p}'_c$ is out of range.

B. Adaptive RANSAC Threshold Decision

A distance threshold for outlier filtering is determined in real-time based on the adaptive RANSAC threshold decision. The proposed approach determines the distance threshold according to environmental changes such as the object shape and the distance scale between the camera and the object. Therefore, unlike the previous setting of the distance threshold offline by a user, it enables continuous updating of the distance threshold in real-time. As a result, the stability of tracking can be improved.

Table 3 summarizes the steps of the adaptive RANSAC threshold decision. Prior to the loop process, the

Table 2. Outlier filtering steps for classifying inliers and outliers.

1) Compute a perspective projection matrix $\mathbf{H}$ from between the features $\mathbf{p}_w$ in the object image and $\mathbf{p}_c$ in the scene image using RANSAC.
2) Compute the Euclidean distance values d_i between the ith feature $\mathbf{p}_{c_i}$ and a transformed feature $\mathbf{p}'_{c_i} = \mathbf{H} \cdot \mathbf{p}_{c_i}$.
3) If d_i is smaller than a case-specific threshold, it is an inlier. In other cases, it is an outlier.

Table 3. Adaptive RANSAC distance threshold decision steps.

1) Set the initial threshold to $t_0 = s/100$, where the scale s is the diagonal length of the object in the camera input image.
2) The Euclidean distances $d_{i,j}$ between $\mathbf{p}_{c_{i,j}}$ and $\mathbf{p}'_{c_{i,j}}$ are computed with the jth feature at every ith frame until the nth frame.
3) At the nth frame, $d_{j\text{mean}} = \frac{1}{n}\sum_{i=1}^{n} d_{i,j}$ is calculated as the sum of each jth feature with all ith frames.
4) If $d_{j\text{mean}}$ is twice $d_{\text{mean}} = \frac{1}{k}\sum_{j=1}^{k} d_{j\text{mean}}$, $d_{j\text{mean}}$ is removed to exclude impulse errors. k is the number of features.
5) Update the distance threshold t_1 through the $d_{j\text{mean}}$ dataset computed using Otsu threshold decision [27] methods. Finally, the normalized threshold $t_{\text{nor}} = t_1/s$ is computed.
6) At every nth frame interval, steps 2) through 5) are repeated, and the normalized distance threshold is updated.

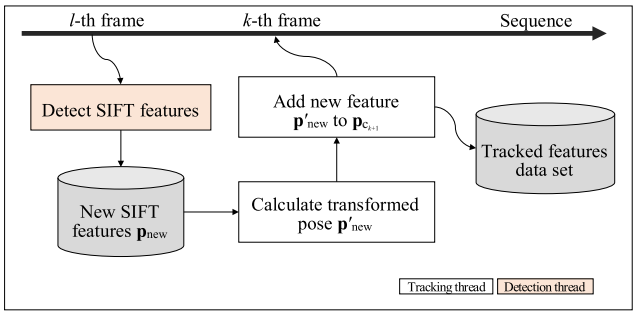


Fig. 6. Flowchart of the detection thread for adding new SIFT features to a tracked feature dataset in the tracking thread.

temporary initial threshold is determined when the initial detected features $\mathbf{p}_c$ are added to the tracking thread. The initial threshold t_0 is set according to the object scale. We set the interval of the threshold decision to 100 frames in our experiment. The adaptive threshold decision is performed every 100 frames in real-time. Here, $t_{\text{nor}} \times s$ is really used as the distance threshold in the outlier filtering. This means that the threshold is adaptively changed according to the distance between the camera and the object.

2. Detection Thread

The detection thread is another thread of the proposed multithreaded object tracker. This thread detects SIFT features in the scene image from the camera input image. However, the SIFT detection method is too slow and is not suitable for a real-time system. In our approach, SIFT detection does not affect the tracking speed of the tracking thread because the detection thread is generated intermittently under certain conditions in a new thread.

Figure 6 shows a flowchart of the detection thread when adding new SIFT features to the tracked feature dataset in the tracking thread. When the detection thread starts at the lth frame, SIFT detection detects new SIFT features in the lth frame. The speed of the SIFT detection procedure is generally 340 ms for detecting and matching about 700 features. For application in a real-time system, all procedures must be performed within 33.3 ms using a 30-fps camera. Thus, newly detected SIFT features cannot be added to the lth frame in real-time.

Newly detected SIFT features must be added to the next few frames later. New features are added to the tracked feature dataset in the tracking thread at the kth frame, as shown in Fig. 6. New features are added to $\mathbf{p}_{c_{k+1}}$, which is described in Table 1. It cannot be assumed that all lth and kth frames are captured from the same camera position. Therefore, we calculate the perspective projection matrix between the tracked features $\mathbf{p}_l$ at the lth frame and $\mathbf{p}_{c_{k+1}}$ at the kth frame.

$$\mathbf{p}_{c_{k+1}} = \mathbf{H}.\mathbf{p}_l. \tag{9}$$

Finally, the transformed pose $\mathbf{p}'_{\text{new}}$ of the newly detected features are computed using $\mathbf{H}$.

$$\mathbf{p}'_{\text{new}} = \mathbf{H}.\mathbf{P}_{\text{new}} \tag{10}$$

when $\mathbf{p}'_{\text{new}}$ is added to the tracked features $\mathbf{p}_{c_{k+1}}$, we need to confirm that the same features exist between them. If any features of $\mathbf{p}'_{\text{new}}$ are the same as one of $\mathbf{p}_{c_{k+1}}$, a feature cannot be added. We calculate the Euclidean distances between one of the $\mathbf{p}'_{\text{new}}$ features and all of $\mathbf{p}_{c_{k+1}}$. If any distance is smaller than a case-specific threshold, we determine that the point is a duplicate.

IV. Experiments

1. Tracking Stability and Speed

We evaluated the tracking experiment using a projector and camera unit for an SAR system through occlusions. An occlusion is referred to as a sudden change in the lighting environment or the generation of an obstacle between the camera and tracked object. The experiment was performed using a desktop computer with an Intel® 2.67-GHz Core™ i7 CPU, a projector (Optoma P320 with a resolution of 1,280 × 720), and a USB 2.0 camera (Logitech C920 with a resolution of 640 × 480).

We compared the tracking stability and speed of the proposed multithreaded object tracker using the SIFT, SURF, or ORB based on a single thread. The experimental video frames were captured using a fixed camera and fixed planar object for calculating the root-mean-square error (RMSE). The ground truth of the fixed camera pose, as a

real camera pose, was computed by measuring the four corner points of a rectangular planar object from the camera input image. Experimental camera poses were calculated through the PnP estimation results of each tracker. A camera pose is the 4th column vector **t** of the extrinsic camera parameter $\mathbf{X}_{wc}$ and is given by

$$\mathbf{p}_c = \mathbf{M}_c \mathbf{X}_{wc} \mathbf{p}_w = \mathbf{M}_c \left[\mathbf{R} \, (x, y, z, 1)^T \right] \mathbf{p}_w. \quad (11)$$

We calculated the RMSEs of the Euclidean distance values between the real camera pose and the camera poses estimated by each tracker. The ideal result is a difference in RMSE of close to zero. Figure 7 shows the first experimental results in an SAR environment through frequent occlusions. The SIFT and proposed method tracked the object more robustly than the SURF or ORB, as shown in Fig. 7(a and b). The processing speed results are shown in Fig. 7(c). The speed results include only the detection, matching, and tracking process times. The SIFT is the slowest approach among those considered. On the other hand, the proposed method is the fastest approach because it is based on a multithreaded system. In conclusion, we found that our proposed tracker is the most effective method in terms of the tracking stability and speed.

In another experiment, we evaluated the pose changes along the *x*, *y*, and *z* axes by moving the camera or object. Although the SIFT is the slowest tracker, it is the most stable approach, as shown in Fig. 8. The proposed tracker shows similar results as the SIFT. In the case of the ORB

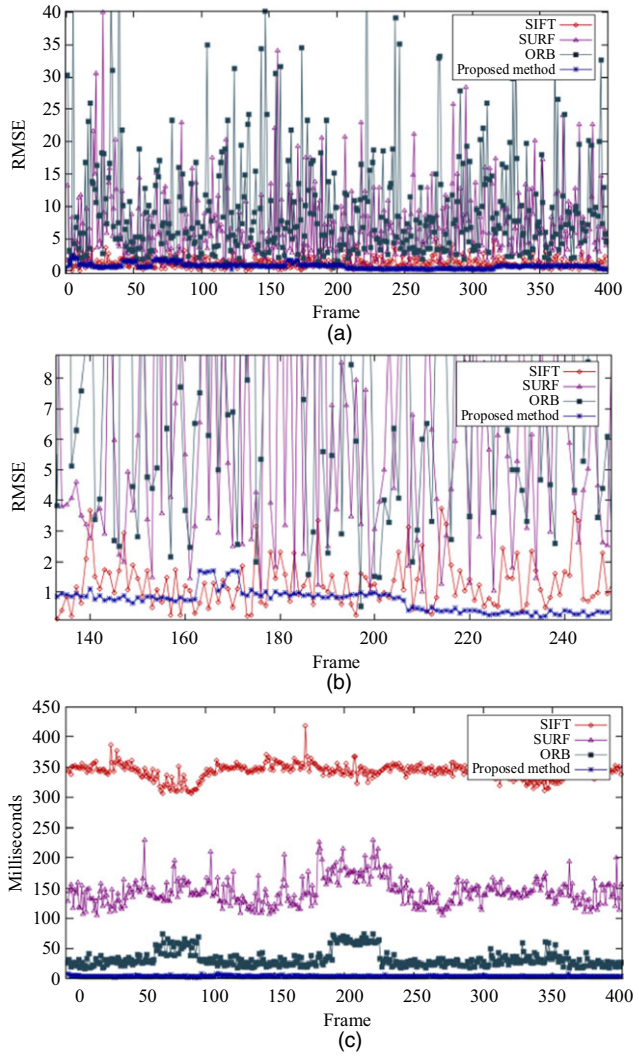


Fig. 7. RMSEs of the Euclidean distances for the SIFT, SURF, ORB, and proposed methods and each processing speed when the camera and planar object are fixed: (a) RMSE between the real camera position and the estimated camera positions, (b) magnified graph of (a), and (c) the processing speed results.

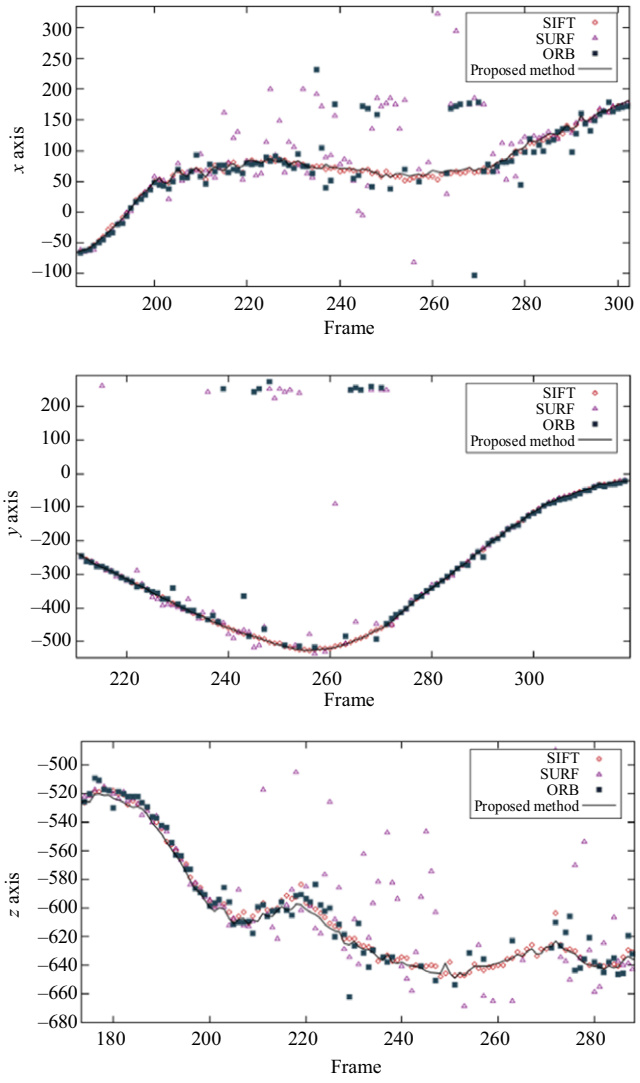


Fig. 8. Tracking results for the camera position using the SIFT, SURF, ORB, and proposed tracker when the camera or projector is moving.

https://doi.org/10.4218/etrij.2017-0047

Table 4. Process times of each step in the tracking thread (400 frames as an average value).

	Time per frame (ms)
Tracking thread	**7.4724**
Track scene features with an LKT	3.2638
Adaptive RANSAC threshold decision	0.3392
Outlier filtering	0.3819
Pose estimation	0.5352
Rendering	2.9523

or SURF, some of the results were not expressed in the graphs with a limited size owing to the presence of overly large errors. Table 4 summarizes the process times of each step in the tracking thread of the proposed multithreaded tracker. The process time results were measured when tracking more than 300 features during 400 frames.

2. Threshold Method for an Adaptive Threshold Decision

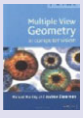
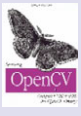
A distance threshold value that classifies inliers and outliers in outlier filtering is an essential factor of the filtering performance. A threshold method for determining the threshold value for dividing $d_{j\text{mean}}$ into 2 was found from the data collected from each jth feature during specific frame intervals, as presented in Table 3. In this study, we attempted to find an effective threshold method for our SAR system [23]–[27]. The experimental video frames were captured using a fixed camera and fixed planar object for calculating the RMSE between the ground truth of the fixed camera pose and the experimental camera poses. Depending on the threshold method used, t_{nor} is determined differently, and the tracking results are also different.

Table 5 summarizes the experiment results for finding an effective threshold method for our SAR system. The value of t_{nor} is very small because it is divided by the diagonal length of the object in the camera coordinates. These experiment results indicate that Otsu's method was the most effective way to determine the adaptive threshold decision for outlier filtering. However, the best way differed according to the objects. We thought that a threshold method should be an optional parameter in the proposed outlier filtering. In general, a low threshold results in a low difference error. However, a low t_{nor} causes a low number of inliers. As a result, a small number of inliers generates a poor pose estimation result. In addition, we compared the results from another multithreaded tracker [8] without using the proposed outlier filtering.

3. Panorama Image Stitching with Real-Time Preview

The proposed tracker enables the robust tracking of a planar object in real-time. Therefore, we evaluated panorama image stitching with a real-time preview using our tracker. Panorama image stitching is a well-known

Table 5. RMSE according to each threshold method using various planar objects. Bold numbers indicate the smallest value in each object.

Threshold decision method	Objects					Average RMSE
Intermodes [23]	t_{nor}	0.0057	0.0034	0.0043	0.0024	N/A
	RMSE	4.7388	3.1944	3.0971	4.1119	3.7856
Iterative [24]	t_{nor}	0.0103	0.0058	0.0043	0.0034	N/A
	RMSE	6.0300	2.4545	3.2684	6.5352	4.5720
Moments [25]	t_{nor}	0.0304	**0.0030**	0.0056	**0.0022**	N/A
	RMSE	15.0666	2.8198	**2.8722**	**3.1803**	5.9847
Percentile [26]	t_{nor}	0.0117	0.0095	0.0073	0.0031	N/A
	RMSE	5.2985	**1.6011**	3.3572	9.3309	4.8769
Otsu [27]	t_{nor}	**0.0055**	0.0037	**0.0022**	0.0048	N/A
	RMSE	**4.2006**	3.1935	4.0536	3.6529	**3.7752**
Without filtering [8]	RMSE	18.8934	11.5217	24.2251	25.9701	20.1526

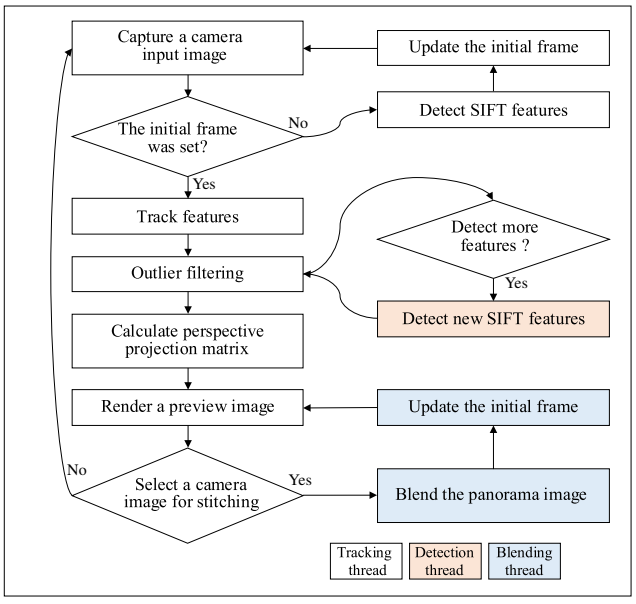


Fig. 9. Workflow of the multithreaded panorama image stitching with real-time preview.

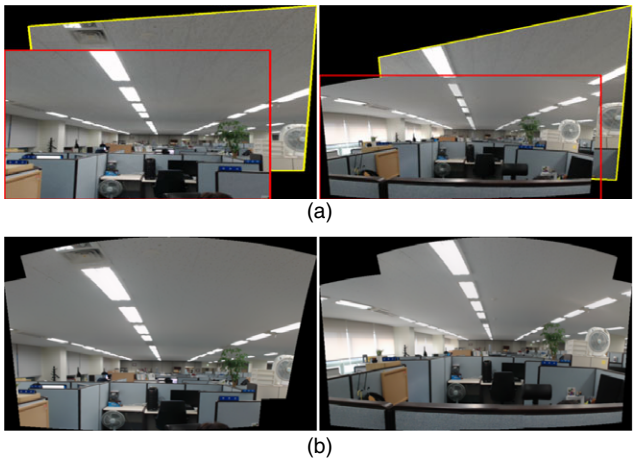


Fig. 10. An example of panorama image stitching: (a) real-time preview results and (b) blended results using the blending thread.

method for creating a wide-angle image with limited tools or environments. Previous approaches have been concerned with the quality of the panorama image stitching result or fully automated systems [28]–[30]. These stitching programs select the source images, while users predict the stitching result using a guideline that helps capture a source image. To create a view of a user-desired shape, trial-and-error is required.

In this paper, we propose a multithreaded panorama image stitching method with a real-time preview. Our panorama image stitching detects and tracks SIFT features in real-time and estimates the perspective projection transformation between the camera input images using the proposed multithreaded tracker. The proposed system is composed of tracking, detection, and blending parts. The blending thread based on an automatic panoramic image stitching method [31] is difficult to execute in real-time; thus, we configured a real-time executable multithreaded system, as shown in Fig. 9.

The initial frame is the object image that should be tracked and detected in the loop process. When a user selects a source image for panorama image stitching, the initial frame is updated as a blending result through the blending thread. SIFT features are detected from the updated initial frame for the later tracking process. Real-time preview rendering is performed based on the tracking result until a user selects an additional source image. Our contribution is the implementation of a user-friendly and real-time-based panorama image stitching system. A real-time preview can be shown in real-time according to the user's controls, as shown in Fig. 10(a). This can help the user easily generate a desired panorama image such as a large land view or building image without trial-and-error.

V. Conclusion

We introduced a robust multithreaded object tracker with outlier filtering for an SAR system. Our proposed tracker is composed of two threads, the tracking and detection threads. Only the tracking thread, as the main thread, is executed in a frame-to-frame manner, and it enables an object to be tracked in real-time. However, the optical-flow-based LKT has unstable tracking results owing to occlusions, which easily remove or deform the tracked features. Our outlier filtering automatically updates the RANSAC distance threshold according to environmental changes. In addition, we consider the most effective threshold decision method for outlier filtering. The experiment results show that our approach with adaptive threshold-decision-based outlier filtering enables a real-world object to be robustly tracked for an SAR that has frequent occlusions from augmented projection images. Moreover, we evaluated a panorama image-stitching system with a real-time preview by applying the proposed multithreaded tracker. We believe that our approach will be a highly practical solution for tracking planar objects in other applications.

Acknowledgements

This work was supported by the Ministry of Land, Infrastructure and Transport (MOLIT), Korea under the Urban Planning & Architecture (UPA) research support

program supervised by the Korea Agency for Infrastructure Technology Advancement (KAIA) (grant 13 Urban Planning & Architecture 02).

References

[1] H. Benko, R. Jota, and A. Wilson, "MirageTable: Freehand Interaction on a Projected Augmented Reality Tabletop," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, Austin, TX, USA, May 2012, pp. 199–208.

[2] J.H. Lee et al., "FRC Based Augment Reality for Aiding Cooperative Activities," in *2013 IEEE RO-MAN*, Gyeongju, Rep. of Korea, Aug. 2013, pp. 294–295.

[3] R. Ziola, S. Grampurohit, N. Landes, J. Fogarty, and B. Harrison, "Examining Interaction with General-Purpose Object Recognition in LEGO OASIS," in *2011 IEEE Symp. Vis. Lang. Human-Centric Comput. (VL/HCC)*, Pittsburgh, PA, USA, Sept. 2011, pp. 65–68.

[4] A. Lee, J.D. Suh, and J. Lee, "Interactive Design of Planar Curves Based on Spatial Augmented Reality," in *Proc. Companion Publication Int. Conf. Intell. User Interfaces Companion*, Santa Monica, CA, USA, Mar. 2013, pp. 53–54.

[5] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, 2004, pp. 91–110.

[6] H. Bay, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Eur. Conf. Comput. Vis.*, vol. 3951, 2006, pp. 404–417.

[7] E. Rublee, T. Tuytelaars, and L. Van Gool, "ORB: An Efficient Alternative to SIFT or SURF," *2011 IEEE Int. Conf. Comput. Vis. (ICCV)*, Barcelona, Spain, Nov. 2011, pp. 2564–2571.

[8] T. Lee and T. Hollerer, "Multithreaded Hybrid Feature Tracking for Markerless Augmented Reality," *IEEE Trans. Vis. Comput. Graph.*, vol. 15, no. 3, 2009, pp. 355–368.

[9] A. Davison, I.D. Reid, N.D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, 2007, pp. 1052–1067.

[10] A. Lee, J.H. Lee, and J. Kim, "Data-Driven Kinematic Control for Robotic Spatial Augmented Reality System with Loose Kinematic Specifications," *ETRI J.*, vol. 38, no. 2, Apr. 2016, pp. 337–346.

[11] X. Wang, Z. Yao, and Z. Yang, "The Use of Object Tracking in Visual SLAM," in *IEEE Int. Conf. Appl. Syst. Innovation (ICASI)*, Sapporo, Japan, May 2017, pp. 850–853.

[12] M.A. Fischler and C.B. Robert, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Commun. ACM*, vol. 24, no. 6, 1981, pp. 381–395.

[13] R. Mur-Artal, J.M.M. Montiel, and J.D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Trans. Robot.*, vol. 31, no. 5, 2015, pp. 1147–1163.

[14] L. Cheng, M. Li, Y. Liu, W. Cai, Y. Chen, and K. Yang, "Remote Sensing Image Matching by Integrating Affine Invariant Feature Extraction and RANSAC," *Comput. Electr. Eng.*, vol. 38, no. 4, 2012, pp. 1023–1032.

[15] A. Lee and J.-H. Lee, "Multi-threaded Tracker with Outlier Filtering for Spatial Augmented Reality," in *Int. Tech. Conf. Circuits Syst., Comput. Commun. (ITC-CSCC)*, Seoul, Rep. of Korea, July 2015, pp. 494–495.

[16] J.-H. Lee et al., "Calibration Issues in FRC: Camera, Projector, Kinematics Based Hybrid Approach," in *Proc. Ubiquitous Robots Ambient Intell.*, Daejeon, Rep. of Korea, Nov. 2012, pp. 218–219.

[17] J.-H. Lee, "An Analytic Solution to Projector Pose Estimation Problem," *ETRI J.*, vol. 34, no. 6, Dec. 2012, pp. 978–981.

[18] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, Nov. 2000, pp. 1330–1334.

[19] J. Weng, P. Cohen, and M. Herniou, "Camera Calibration with Distortion Models and Accuracy Evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 10, 1992, pp. 965–980.

[20] R. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," *IEEE J. Robot. Autom.*, vol. 3, no. 4, 1987, pp. 323–344.

[21] J.-Y. Bouguet, "Pyramidal Implementation of the Affine Lucas Kanade Feature Tracker Description of the Algorithm," *Intel Corp.*, vol. 5, 2001, pp. 1–10.

[22] J.-W. Choi, D. Moon, and H.H. Yoo, "Robust Multi-person Tracking for Real-Time Intelligent Video Surveillance," *ETRI J.*, vol. 37, no. 3, June 2015, pp. 551–561.

[23] J.M. Prewitt and M.L. Mendelsohn, "The Analysis of Cell Images," *Annu. NY Acad. Sci*, vol. 128, no. 3, 1966, pp. 1035–1053.

[24] M. Fornasier and H. Rauhut, "Iterative Thresholding Algorithms," *Appl. Comput. Harmon. Anal.*, vol. 25, no. 2, 2008, pp. 187–208.

[25] W.H. Tsai, "Moment-Preserving Thresholding: A New Approach," *Comput. Vis. Graph. Image Process.*, vol. 29, no. 3, 1985, pp. 377–393.

[26] W. Doyle, "Operations Useful for Similarity-Invariant Pattern Recognition," *J. ACM*, vol. 9, no. 2, 1962, pp. 259–267.

[27] N. Otsu, “A Threshold Selection Method from Gray-Level Histograms,” *IEEE Trans. Sys. Man. Cybern.*, vol. 9, 1975, pp. 62–66.

[28] J.H. Cha, Y.S. Jeon, Y.S. Moon, and S.H. Lee, “Seamless and Fast Panoramic Image Stitching,” in *IEEE Int. Conf. Consumer Electron. (ICCE)*, Las Vegas, NV, USA, Jan. 2012, pp. 29–30.

[29] F. Zhang and F. Liu, “Parallax-Tolerant Image Stitching,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Columbus, OH, USA, June 2014, pp. 3262–3269.

[30] Y. Xiong and K. Pulli, “Sequential Image Stitching for Nobile Panoramas,” in *IEEE Int. Conf. Inf., Commun. Signal Process. (ICICS)*, Macau, China, Dec. 2009.

[31] M. Brown and D.G. Lowe, “Automatic Panoramic Image Stitching Using Invariant Features,” *Int. J. Comput. Vis.*, vol. 74, no. 1, 2007, pp. 59–73.

Ahyun Lee received his PhD degree in computer science from the University of Science & Technology, Daejeon, Rep. of Korea. From 2011 to 2012, he was an engineer at LG Electronics Inc. Pyeongtaek, Rep. of Korea. He has been a research scientist at ETRI, Daejeon, Rep. of Korea since 2016. His research interests include computer vision, augmented reality, robotics, and spatial information.

Insung Jang received BS and MS degrees in computer engineering from Pusan National University, Rep. of Korea in 1999 and 2001, respectively. Since 2001, he has been a senior member of the research staff at ETRI, Daejeon, Rep. of Korea, and he is also working toward a PhD degree in computer engineering from Pusan National University. His main research areas are platforms for geosensor service, navigation service, and location-based service.

https://doi.org/10.4218/etrij.2017-0047