

A Novel Interaction Method for Mobile Devices Using Low Complexity Global Motion Estimation

Toan Dinh Nguyen, JeongHwan Kim, SooHyung Kim, HyungJeong Yang, GueeSang Lee, JuneYoung Chang, and NakWoong Eum

A novel interaction method for mobile phones using their built-in cameras is presented. By estimating the path connecting the center points of frames captured by the camera phone, objects of interest can be easily extracted and recognized. To estimate the movement of the mobile phone, corners and corresponding Speeded-Up Robust Features descriptors are used to calculate the spatial transformation parameters between the previous and current frames. These parameters are then used to recalculate the locations of the center points in the previous frame into the current frame. The experiment results obtained from real image sequences show that the proposed system is efficient, flexible, and able to provide accurate and stable results.

Keywords: Interaction method, global motion estimation, mobile device, SURF, corner detection.

I. Introduction

The mobile phone is now the most widely used communication technology device in the world. With the diversity of applications adopted for the mobile phone, new interaction methods should be developed to cope with the limitation of the interface due to its small size. Sensors embedded in the device can be used for physical manipulation, such as contact, pressure, tilt, and implicit biometric information. The tilt input can be used for navigating menus, maps, and 3-D scenes or scrolling through documents and lists [1]-[3]. Another interaction method is based on the motion of mobile phones. Estimation of device motion often makes use of motion sensors, such as accelerometers [3], [4]. However, in this case, an external sensor needs to be installed. Since computer vision is a more natural choice, we focus herein on using computer vision in interaction systems. Previous studies cover multimodal interaction [5], gesture recognition, face tracking, and body tracking [6]. However, most of these methods are built on powerful desktop computers.

In TinyMotion [7], [8], image sequences captured by built-in cameras are analyzed in real time, without requiring additional sensors. TinyMotion is based on both image differencing and correlation of blocks for motion estimation (ME) and can efficiently detect horizontal, vertical, rotational, and tilt movements. However, this method detects only the moving directions (up, down, left, and right) and does not provide information about the location of the camera and the distance of the movement. Another approach to estimate camera motion is to use the global ME (GME) technique, which has been widely used in many applications, such as video coding, video stabilization, content-based video analysis, and motion

Manuscript received Jan. 31, 2012; revised May 23, 2012; accepted June 22, 2012.

This work was supported by the Basic Science Research Program through the National Research Fund of Korea (NRF), funded by the Ministry of Education, Science and Technology (2011-0006109) and (2011-0029429).

Toan Dinh Nguyen (+82 62 530 3425, toan_mulmi@hotmail.com), JeongHwan Kim (disturb@naver.com), SooHyung Kim (shkim@jnu.ac.kr), HyungJeong Yang (hjiyang@jnu.ac.kr), and GueeSang Lee (corresponding author, gslee@jnu.ac.kr) are with the Department of Electronics and Computer Engineering, Chonnam National University, Gwangju, Rep. of Korea.

JuneYoung Chang (jychang@etri.re.kr) is with the Creative & Challenging Research Division, ETRI, Daejeon, Rep. of Korea.

NakWoong Eum (nweum@etri.re.kr) is with the SW-SoC Convergence Research Laboratory, ETRI, Daejeon, Rep. of Korea.

<http://dx.doi.org/10.4218/etrij.12.0112.0070>

segmentation. To define the motion field of global motion, motion models are used with corresponding parameters: translational (two parameters), geometric (four parameters), affine (six parameters), and perspective (eight parameters). GME can be done in the pixel domain [9] by solving a set of linear equations with the help of feature points correspondence. In [10], an interface based on motion for control of mobile devices is proposed. In this method, mobile functions, defined by a series of motions, are carried out by the movement of the mobile device, using hidden Markov models (HMMs). The features for the HMMs are estimated using the global motion of the mobile phone. This process includes two phases: 1) the motion of the selected features and related uncertainty is analyzed, and 2) outliers are removed and the results are used for obtaining the parameters of the global motion. However, feature generation can be very time-consuming. Furthermore, since the gradient feature is based on the assumption of the texture/gradient distribution of the surrounding environments, it may fail when this assumption does not hold. The motion model can only capture the translation between two frames, and it is therefore not good enough to find the path of the camera movement.

In this paper, a new interaction method for mobile phones is developed by analyzing the image sequences captured from built-in cameras. The movement of the center point or any interest point in the image sequence is obtained according to the camera motion. In other words, a user can specify a path containing a series of center points in the scene by moving the camera. This technique can be used in several real applications; for example, a user can specify a contour around an object to segment and find information about it (Fig. 1 (a)) or to indicate a curled text line to extract it from the scene (Fig. 1 (b)).

The main challenges in this interaction technique are as follows: 1) The proposed method needs to process and display the intermediate path in real time. To cope with this problem, corners are used as feature points with Speeded-Up Robust Features (SURF) descriptors instead of Scale-Invariant Feature Transform (SIFT) descriptors. 2) The image sequence captured from the camera movement may include several types of transformation and distortion. The speed of the movement is also unpredictable. For these reasons, block ME may not be applicable to finding the movement of the center point. Therefore, a low complexity GME method must be developed to recalculate the locations of center points.

In our previous approach [11], SURF feature points and SURF descriptors are used to calculate the spatial transformation parameters to map center points in the previous frame into the current frame. ME is then used to refine the locations of the center points in the current frame. Compared to this work, the proposed method, CORNER-SURF, contains

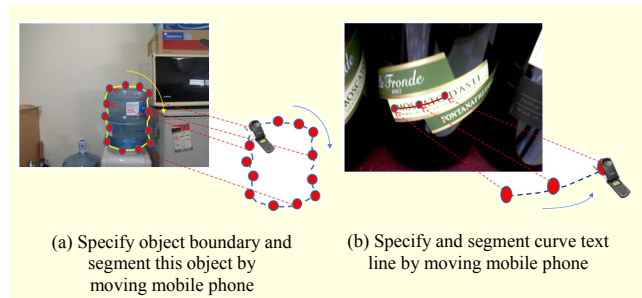


Fig. 1. Some target application of our interaction technique.

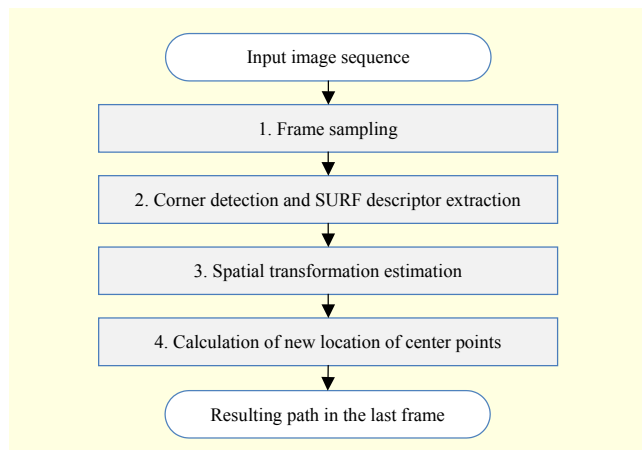


Fig. 2. Flowchart for proposed method.

several improvements: 1) corners are used instead of SURF feature points, and local motion vectors are found by feature matching instead of local ME to reduce the computation time. Note that corner features can be generated much easier than SURF features with scale invariance, which may not be required in image sequences like video data. 2) Extended results are generated, compared to the results of other methods. CORNER-SURF can provide better results with lower complexity, compared to the results provided by the work in [11].

II. Proposed Interaction Technique

1. Overview of Proposed Method

This proposed interaction method is based on the path connecting the center points of frames in the image sequence captured by the built-in camera, according to the movement of the mobile phone. In other words, a user can use the center points of the frame images to indicate the path, such as the contour of an object or a curled text line. A flowchart of CORNER-SURF is shown in Fig. 2. First, the input frames are sampled to reduce the complexity by reducing the number of frames that need to be processed. The low complexity Harris

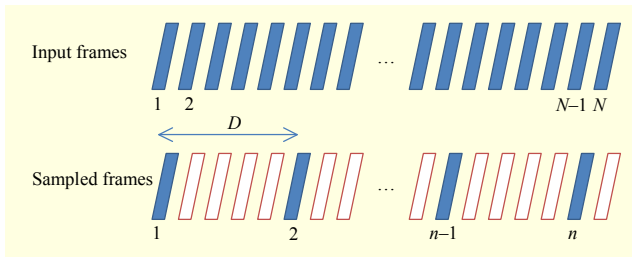


Fig. 3. Frame sampling ($D=5$).

corner detector [12] is used to detect the corners in the current frame. The SURF descriptors [13] are then calculated at the corners to find the corresponding corner pairs between previous and current sampled frames. By using the corner point correspondence obtained from the previous step, the spatial transformation parameters that map these two frames are estimated. Finally, the locations of all center points in previous frames are recalculated using the obtained spatial transformation parameters. In the last frame, the path is defined as a curve connecting all center points of frames together.

2. Reducing Number of Frames

To reduce the number of frames, the input image sequence is sampled by frame sampling. The distance between two sampled frames is set to the value D , as shown in Fig. 3. To reduce the complexity for processing the whole image sequence, the path connecting all frame center points in the image sequence is approximated by using the sampled frames. When a smooth path is needed, a Bezier curve, which connects the center points of the sampled frames obtained in the last frame, is calculated. Furthermore, the spatial transformation parameters can be estimated efficiently when the difference between two successively sampled frames is big enough. By processing only the sampled frames, the processing time for each frame is extended, and CORNER-SURF can therefore process in real time. The resolution of the sampled frames can then be downsampled to reduce the computation complexity, if necessary. At the end of this stage, the sampled frames are converted into grayscale images.

3. Corner Detection for Feature Point Generation

To estimate the global motion between sampled frames, the pairs of corresponding feature points are first estimated. Instead of using SURF feature points as in [11], the corners detected by the low complexity Harris corner detector [12] are used. In the Harris corner detector, corners are directly detected using the differential of the corner score with respect to direction. This detector is based on the local autocorrelation function, which measures local changes of the image, with patches shifted by a

small amount in different directions. Given a shift $(\Delta x, \Delta y)$ and a point (x, y) , the autocorrelation function is defined as

$$S(x, y) = \sum_W [I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y)]^2, \quad (1)$$

where (x_i, y_i) are the points in the (Gaussian) window W centered on (x, y) . A Taylor expansion is then used to approximate the shifted image by truncating to the first order terms:

$$I(x_i + \Delta x, y_i + \Delta y) \approx I(x_i, y_i) + [I_x(x_i, y_i) \quad I_y(x_i, y_i)] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}, \quad (2)$$

where I_x and I_y are the partial derivatives of the grayscale image I along the x and y axes, respectively.

Substituting (2) into (1) yields

$$S(x, y) = \sum_W ([I_x(x_i, y_i) \quad I_y(x_i, y_i)] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix})^2, \quad (3)$$

$$\begin{aligned} S(x, y) &= [\Delta x \quad \Delta y] \begin{bmatrix} \sum_W (I_x(x_i, y_i))^2 & \sum_W I_x(x_i, y_i) I_y(x_i, y_i) \\ \sum_W I_x(x_i, y_i) I_y(x_i, y_i) & \sum_W (I_y(x_i, y_i))^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}, \end{aligned} \quad (4)$$

$$S(x, y) = [\Delta x \quad \Delta y] C(x, y) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}, \quad (5)$$

where $C(x, y)$ is a structure tensor (Harris matrix) that captures the intensity structure of the local neighborhood. Let us denote λ_1 and λ_2 as eigenvalues of the Harris matrix $C(x, y)$. There are three cases to be considered [12]:

- Case 1: If both λ_1 and λ_2 are small, the local autocorrelation function is flat, and, therefore, the windowed image region is of approximately constant intensity.
- Case 2: If $\lambda_1 \gg \lambda_2$ or $\lambda_2 \gg \lambda_1$, the local autocorrelation function is ridge shaped, and, therefore, there is an edge at (x, y) .
- Case 3: If $\lambda_1 \gg 0$ and $\lambda_2 \gg 0$, then the local autocorrelation function is sharply peaked, and, therefore, it indicates a corner.

The value of $R(x, y)$, which is used to determine whether the pixel at (x, y) is a corner or not, is calculated as follows:

$$\text{Det}(C(x, y)) = \lambda_1 + \lambda_2, \quad (6)$$

$$\text{Tr}(C(x, y)) = \lambda_1 \lambda_2, \quad (7)$$

$$R(x, y) = \text{Det}(C(x, y)) - k \text{Tr}^2(C(x, y)), \quad (8)$$

where $R(x, y)$ is positive in the corner regions, negative in the edge regions, and small in value in the flat regions. Thus, if

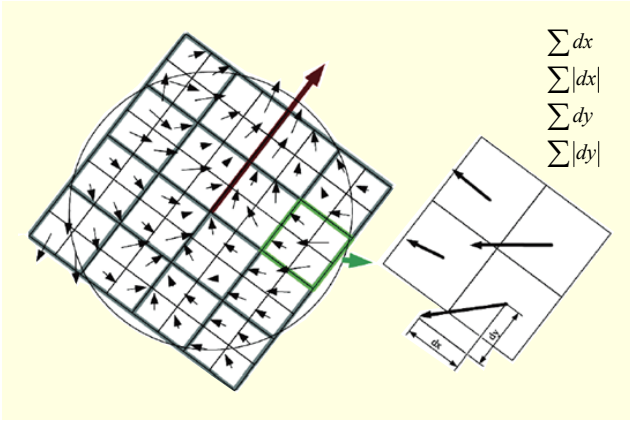


Fig. 4. Calculation of SURF descriptor by aligning oriented quadratic grid at SURF feature point [13].

$R(x, y)$ is positive and maximum in the 8-adjacent, there is a corner at (x, y) .

4. SURF Descriptors with Corner Features

SURF [13] is stable with changes of scale, illumination, and rotation, and partially stable with affine invariance. Therefore, it is suitable for estimating the spatial transformation between the previous and current sampled frames. The SURF descriptor is used in this work, because it can generate good results compared to those of the SIFT descriptor [14], with much lower computational complexity.

In the sampled image sequence, let us denote the current frame as F_n and the reference (previous) frame as F_{n-1} . Let us denote the two sets of corners of the current frame, Φ_n , and reference frame, Φ_{n-1} , as

$$\Phi_n = \{(x_1, y_1), (x_2, y_2), \dots, (x_{m_1}, y_{m_1})\}, \quad (9)$$

$$\Phi_{n-1} = \{(x'_1, y'_1), (x'_2, y'_2), \dots, (x'_{m_2}, y'_{m_2})\}, \quad (10)$$

where m_1 and m_2 are the number of corners detected in the current and reference frames. The two sets of SURF descriptors for the corners are defined as

$$\Omega_n = \{\omega_1, \omega_2, \dots, \omega_{m_1}\}, \quad (11)$$

$$\Omega_{n-1} = \{\omega'_1, \omega'_2, \dots, \omega'_{m_2}\}, \quad (12)$$

where ω_i ($i=1, \dots, m_1$) and ω'_j ($j=1, \dots, m_2$) are the descriptors related to corners at (x_i, y_i) and (x'_j, y'_j) , respectively. The SURF descriptors are calculated using the neighborhood of the corners. To calculate the SURF descriptor for a SURF feature point that is presented by a triple $[(x_i, y_i), s_i, \alpha_i]$ as its location, scale, and orientation, a square region is then centered and aligned along the orientation α_i (Fig. 4). The size of the region is $20s$, where s is the space scale. This region is split into 4×4

subregions. The Haar wavelets are computed in horizontal direction dx and vertical direction dy . The wavelet responses are summed up for each subregion to form the first set of the feature vector. The sums of absolute values of the responses $|dx|$ and $|dy|$ are also added to this vector. Thus, four elements are represented for a subregion. Hence, the descriptor for each feature point is a $4 \times 4 \times 4 = 64$ dimensional vector, compared to a 128 dimensional vector in SIFT. Since the descriptor is computed with respect to the dominant orientation, invariance to rotation is achieved. The descriptor is then normalized to improve the invariance to changes in illumination. Herein, because we use corners as feature points, each corner at (x_i, y_i) can be considered as a SURF feature point $[(x_i, y_i), 1, 0]$.

5. Feature Matching and Global Motion Estimation

When SURF descriptors are generated from the feature points extracted by corner detection, a matching process is applied to find corresponding pairs of the corners in the current and reference frames. In other words, we need to find the matches: $\{\omega_i \leftrightarrow \omega'_j\}, i = 1, \dots, m_1$ and $j = 1, \dots, m_2$.

For matching, the cosine similarity function is adopted, and the 2-Nearest Neighbor (2-NN) search strategy is used for each descriptor ω_i in the set Ω_n to find the two most similar descriptors ω'_j and ω'_k in the set Ω_{n-1} (ω_i is more similar to ω'_j than it is to ω'_k). The two corners, ω_i and ω'_j , become a pair of corresponding points, if the following condition is satisfied:

$$\begin{aligned} v_1 &< DR * v_2, \\ v_1 &= \cos^{-1}(\omega_i * \omega'^T_j), \\ v_2 &= \cos^{-1}(\omega_i * \omega'^T_k), \end{aligned} \quad (13)$$

where the DR is set to 0.6 by empirical selection. Thus, a match is accepted only if the ratio of vector angles from the nearest to the second nearest neighbor is less than DR . Figure 5 shows the matching pairs of feature points between the current and reference frames.

The motion parameters for the global motion are estimated by the spatial transformation matrix that can be calculated using the matching pairs. The nonreflective similarity transformation model, which can include a translation, a rotation, and a scaling, is used. This nonreflective similarity transformation model can be represented in a matrix, which can map every point in the reference frame into the current frame. This model is simple and contains enough transformation information between the two frames. Compared to the affine transformation model, the shapes and angles are preserved in this model. To cope with the small distortion in the shapes, an ME process with small search range is applied herein as a post-processing step. Since we only need two

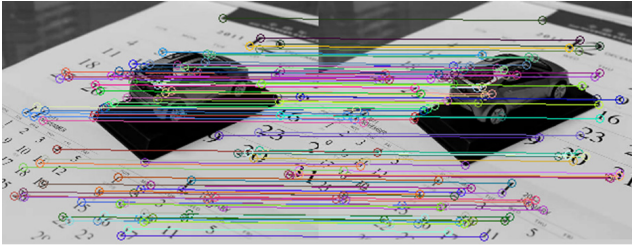


Fig. 5. Matching pairs of corners between current frame and previous frame.

parameters to represent the nonreflective similarity transformation model, at least two matching pairs are required. Let us denote $sc=scale*\cos(angle)$ and $ss=scale*\sin(angle)$. Translations in x and y directions are represented by tx and ty , respectively. The parameters can be easily estimated using a linear system, which is given by

$$[x \ y] = [x' \ y' \ 1] * T = [x' \ y' \ 1] * \begin{bmatrix} sc & -ss \\ ss & sc \\ tx & ty \end{bmatrix}. \quad (14)$$

We can solve for four unknowns, sc , ss , tx , and ty , using the matching pairs. The above equation can be rewritten as

$$x = [x' \ y' \ 1 \ 0] * \begin{bmatrix} sc \\ ss \\ tx \\ ty \end{bmatrix}, \quad (15)$$

$$y = [y' \ -x' \ 1 \ 0] * \begin{bmatrix} sc \\ ss \\ tx \\ ty \end{bmatrix}. \quad (16)$$

With at least two matches, we can combine the x and y equations for one linear system to solve the four unknowns:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \\ y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} x'_1 & y'_1 & 1 & 0 \\ x'_2 & y'_2 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ x'_m & y'_m & 1 & 0 \\ y'_1 & -x'_1 & 1 & 0 \\ y'_2 & -x'_2 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ y'_m & -x'_m & 1 & 0 \end{bmatrix} * \begin{bmatrix} sc \\ ss \\ tx \\ ty \end{bmatrix}. \quad (17)$$

Otherwise, we can rewrite the above matrix equation $X=X'*r$, where $r = [sc \ ss \ tx \ ty]'$, and, therefore, we have $r = X / X'$.

6. Path Extraction

The transformation matrix T maps pixels in the reference

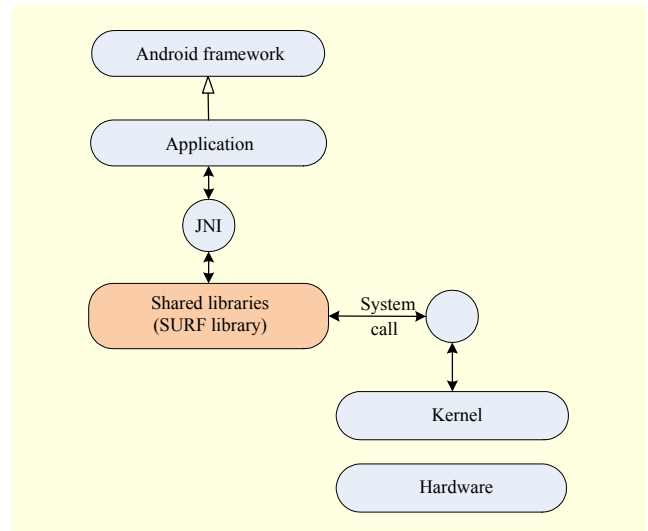


Fig. 6. Flowchart of Android application using shared libraries.

(previous) frame into the current frame. The positions of all center points in the previous sampled frame, F_{n-1} , are recalculated in the current frame, F_n , by using the transformation matrix, T . Because of the camera movement, the center points of previous frames are relocated at the new locations in the current frame, corresponding to the transformation matrix T . After recalculating the locations for all previous center points, the center point of the current frame is added into the path. The path connecting all center points in the last frame is the path that needs to be found.

7. SURF Implementation on Android Mobile Phone

Most of the computational time of CORNER-SURF is spent on calculating SURF descriptors for corners. The main programming language for Android mobile phones using Android SDK is Java. Java applications are compiled to Java bytecode by Java compiler. The Java bytecode is then compiled into .dex files, which can be run by Dalvik virtual machine (Dalvik VM) [15]. The execution time for Dalvik bytecode in .dex files is slow. Therefore, we need to reduce the execution time of the Dalvik VM. For this purpose, SURF descriptor extraction is implemented and built as a native C/C++ library, using the Android native development kit (Android NDK). The Android NDK was developed to improve processing speed, reuse available C source code, and access hardware directly. This SURF library is then dynamically loaded and communicates with Java using the Java Native Interface (Java JNI). The Android NDK can be used with other libraries, such as OpenCV [16]. Hence, we can use the SURF implementation that is already implemented in OpenCV. A flowchart of running an application with shared

libraries on Android is depicted in Fig. 6. Using this approach, the computation for the SURF descriptor extraction turns out to be twice as fast as that of the Java implementation.

III. Experiment Results and Analysis

Several test image sequences captured by built-in mobile phone cameras are used to evaluate the performance of CORNER-SURF. The images are captured with a resolution of 640×480 at 30 frames per second. Three test image sequences having 245 frames (~8 seconds), 265 frames (~8.8 seconds), and 189 frames (~6.3 seconds) are shown in this paper. The algorithm is implemented and tested on an Android mobile phone with a 1 GHz processor and 512 MB of RAM. CORNER-SURF is compared with other methods:

- SURF-ME: This is our previous version [11]. In this algorithm, SURF feature points, SURF descriptors, and ME are used.
- CORNER-ME: Corners and ME are used to find correspondence feature points between two frames. In this algorithm, the corners are first detected by the Harris corner detector. Instead of finding the pairs of corresponding points using SURF descriptors, an ME process is used. For this purpose, a block-based ME process is performed in a small search range for each corner in the current frame to find a corresponding corner in the previous frame. Note that to reduce the complexity of the ME process in the previous frame, we only consider pixels around corners. The block size is set to 32. Finally, the global motion parameters are estimated, as in CORNER-SURF.

To evaluate the accuracy of algorithms, the resulting paths are compared with the ground truth (GT) paths. To extract the GT path for each image sequence, four points, which correspond to the same locations in the whole image sequence, are manually marked. The global motion is then estimated using the perspective (eight parameters) motion model by solving a set of linear equations with the help of selected points correspondence (Fig. 7) [9]. Tables 1 and 2 show the comparison of resulting paths and running time of the methods. The CORNER-ME method is the fastest one, but it generates the worst results. This method generates very bad results when the distance between two sampled D is increased. When $D=30$, the result contains erroneous paths and the result with D values that are too large is unacceptable. The resulting paths obtained by the SURF-ME method are almost the same as the GT. However, the SURF-ME method is time-consuming because the SURF feature points detection is very complex. CORNER-SURF generates good results compared to those of SURF-ME and runs very fast. Thus, CORNER-SURF provides a good

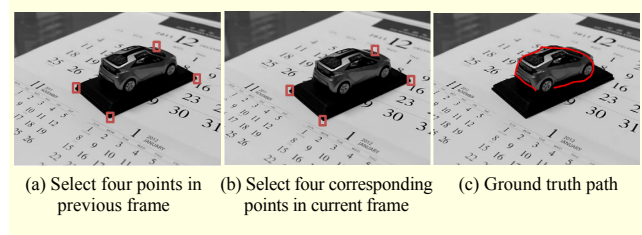


Fig. 7. Ground truth path extraction.

tradeoff between accuracy and computational requirement. The real-time constraint can always be satisfied using the proposed method. Instead of using a fixed value of D , an adaptive approach can be used such that, after processing a frame, the current incoming frame of the image sequence is used as the current frame.

IV. Conclusion

A novel method is proposed to extract the path connecting center points in image sequences, using corner detection and the SURF descriptor. The resulting paths are useful for many applications, such as navigation, object segmentation, and curled text segmentation. The proposed method is simple yet efficient and applicable. The experiment results show that the proposed method is fast and flexible and can be used as an interaction technique for mobile phones [17].

References

- [1] J. Rekimoto, "Tilting Operations for Small Screen Interfaces," *9th Annual ACM Symp. User Interface Softw. Technol.*, Seattle, WA, USA, 1996, pp. 167-168.
- [2] B.L. Harrison et al., "Squeeze Me, Hold Me, Tilt Me! An Exploration of Manipulative User Interfaces," *SIGCHI Conf. Human Factors Computing Systems*, Los Angeles, CA, USA, 1998, pp. 17-24.
- [3] K. Hinckley et al., "Sensing Techniques for Mobile Interaction," *13th Annual ACM Symp. User Interface Softw. Technol.*, San Diego, CA, USA, 2000, pp. 91-100.
- [4] K. Partridge et al., "TiltType: Accelerometer-Supported Text Entry for Very Small Devices," *15th Annual ACM Symp. User Interface Softw. Technol.*, Paris, France, 2002, pp. 201-204.
- [5] H. Yong, J. Back, and T.-J. Jang, "Stereo-Vision-Based Human-Computer Interaction with Tactile Stimulation," *ETRI J.*, vol. 29, no. 3, 2007, pp. 305-310.
- [6] T.B. Moeslund and E. Granum, "A Survey of Computer Vision-Based Human Motion Capture," *Comput. Vis. Image Underst.*, vol. 81, no. 3, 2001, pp. 231-268.
- [7] J. Wang, S. Zhai, and J. Canny, "Camera Phone Based Motion Sensing: Interaction Techniques, Applications and Performance

Table 1. Comparison of path extraction methods.

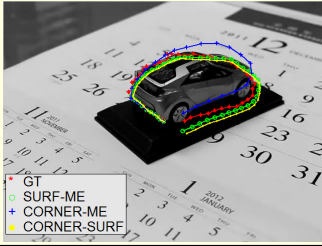
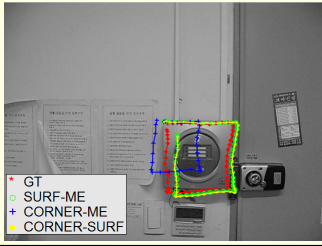
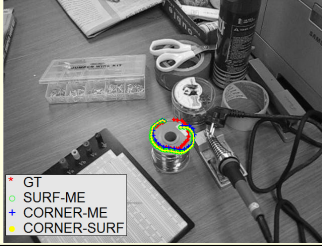

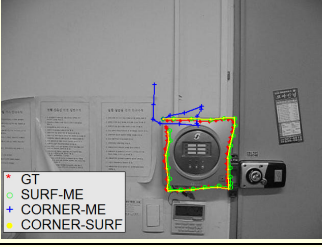

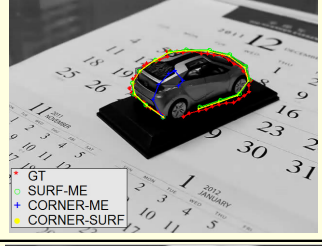
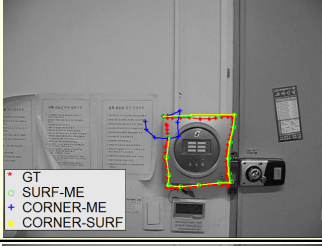


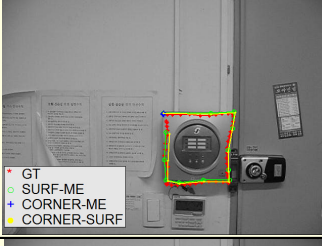


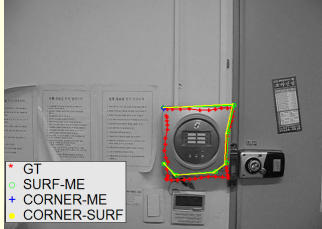

D	Sequence 1 (S1)	Sequence 2 (S2)	Sequence 3 (S3)
5			
10			
15			
20			
30			

Table 2. Comparison of average processing time for whole sequence (seconds).

D	CORNER-ME			SURF-ME			CORNER-SURF		
	S1	S2	S3	S1	S2	S3	S1	S2	S3
5	5.03	5.95	4.54	54.81	43.04	50.33	18.50	16.71	15.57
10	2.23	2.58	2.06	25.39	20.27	23.19	9.20	7.84	6.92
15	1.38	1.76	1.40	16.41	12.91	14.44	6.04	5.38	4.43
20	0.70	0.79	0.48	12.70	11.03	12.31	2.03	2.25	1.38
30	0.46	0.47	0.27	7.68	6.54	7.08	1.26	1.25	0.80

Study,” *19th Annual ACM Symp. User Interface Softw. Technol.*, Montreux, Switzerland, 2006, pp. 101-110.

- [8] J. Wang and J. Canny, “TinyMotion: Camera Phone Based Interaction Methods,” *CHI Extended Abstracts on Human Factors in Computing Systems*, Canada, 2006, pp. 339-344.
- [9] A. Krutz et al., “Windowed Image Registration for Robust Mosaicing of Scenes with Large Background Occlusions,” *ICIP*, 2006, pp. 353-356.
- [10] M. Barnard et al., “A Vision Based Motion Interface for Mobile Phones,” *Computer Vision Syst.*, 2007.
- [11] T.N. Dinh, J.H. Kim, and G. Lee, “Extracting the Path of Frame Center Points Using Spatial Transformation and Motion Estimation,” *Int. Conf. Computer Inf. Technol. (CIT)*, 2011, pp. 283-290.
- [12] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” *4th Alvey Vision Con.*, 1988, pp. 147-151.
- [13] H. Bay et al., “Speeded Up Robust Features (SURF),” *Computer Vision Image Understanding (CVIU)*, vol. 110, 2008, pp. 346-359.
- [14] D.G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *Int. J. Comput. Vision*, vol. 60, 2004, pp. 91-110.
- [15] R. Rogers and J. Lombardo, *Android Application Development*, 1st ed., Sebastopol, CA: O’Reilly Media, Inc., 2009.
- [16] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s J. Softw. Tools*, 2000.
- [17] X. Liang, S. Zhang, and W. Geng, “Motion-Based Perceptual User Interface,” *Third Int. Symp. Intell. Inf. Technol. Appl.*, 2009, pp. 247-251.



Toan Dinh Nguyen received his MS and PhD in computer science from Chonnam National University, Gwangju, Rep. of Korea, in 2008 and 2012, respectively. He is currently an imaging informatics officer of Monash Biomedical Imaging and Monash e-Research Center, Monash University, Melbourne, Australia. His research interests are video coding, text image processing, tensor voting, and software engineering.



JeongHwan Kim received his BS in electronics and computer engineering from Chonnam National University, Gwangju, Rep. of Korea. Currently, he is pursuing his MS in electronics and computer engineering at Chonnam National University. His main research interests are image feature analysis, tensor voting, and mobile applications.



SooHyung Kim received his BS in computer engineering from Seoul National University, Seoul, Rep. of Korea, in 1986 and his MS and PhD in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 1988 and 1993, respectively. From 1990 to 1996, he was a senior member of the research staff in the Multimedia Research Center of Samsung Electronics Co., Rep. of Korea. Since 1997, he has been a professor in the Department of Computer Science, Chonnam National University, Gwangju, Rep. of Korea. His research interests are pattern recognition, document image processing, medical image processing, and ubiquitous computing.



HyungJeong Yang is an associate professor in the Division of Electronics and Computer Engineering at Chonnam National University, Gwangju, Rep. of Korea. Dr. Yang received her BSc, MSc, and PhD in computer science and statistics from Chonbuk National University, Jeonju, Rep. of Korea. She has been a post-doctoral researcher at the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA. Her main research interests include multimedia data mining, e-design, and information retrieval.



GueeSang Lee received his BS in electrical engineering and his MS in computer engineering from Seoul National University, Seoul, Rep. of Korea, in 1980 and 1982, respectively. He received his PhD in computer science from Pennsylvania State University, University Park, PA, USA, in 1991. He is currently a professor of the Department of Electronics and Computer Engineering at Chonnam National University, Gwangju, Rep. of Korea. His main research interests are image processing, computer vision, and video technology.



JuneYoung Chang received his BS in electrical engineering from Chonnam National University, Gwangju, Rep. of Korea, in 1985 and his MS in computer engineering from Chungang University, Seoul, Rep. of Korea, in 1996, respectively. He received his PhD in computer science from Chonnam National University in 1996. He is currently a researcher in ETRI, Daejeon, Rep. of Korea. His main research interests are multi-core platform design, multimedia systems, SoC design, network-on-chip design, and embedded processor design.



NakWoong Eum received his BS in electrical engineering from Kyungpook National University, Daegu, Rep. of Korea, in 1984 and his MS in computer engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 1987, respectively. He received his PhD in computer

science from KAIST, Daejeon, Rep. of Korea, in 2001. He is currently a researcher in ETRI, Daejeon, Rep. of Korea. His main research interests are embedded processor design, mobile multimedia systems, and SoC design.