Thai Alphabet Recognition from Hand Motion Trajectory Using HMM

Kittasil Silanon and Nikom Suvonvorn

***Abstract*—In this paper, we propose a system for Thai alphabet recognition from hand movement trajectory, as a human-computer interaction method. Thai characters drawing by hand movement are analyzed and recognized by our system, which can apply for controlling any specific tasks. The method is based on hand motion analysis combining with Haar-like with a cascade of boost classifiers, as hand detection method. Hand is tracked with skin color using CamShift and Kalman filter. Trajectory features of hand are extracted and used for recognizing 12 Thai alphabet letters though the Hidden Markov Model.**

***Index Terms*—Thai alphabet, hand movement trajectory recognition, hidden markov model**

I. Introduction

In the field of computer vision, communication between human and computer becomes more important. Human-Computer Interaction (HCI) [1], which allows humans communicate with computer have been a popular research field for many years. Some active research in this field are human face recognition, eye gaze tracking, lip reading, hand gesture recognition and body pose tracking. In this paper, we emphasize on hand movement, which can freely moves and gesticulates more than other parts of body therefore hand movement gesture recognition can be applied in many applications such as sign language recognition, computer-controlled game, teleconference, and so on. Many researches of hand gesture recognition have been proposed, covering a wide variety of methods and approaches. For example, Wei Du et. al. [1] describes a system with one camera that can recognize four gestures and tracking hand by extracting the feature points on hand contour. Mahmoud et. al. [2] has developed a system that could recognize gesture for alphabets from hand motion using Hidden Markov Model (HMM). Juan et. al. [3] introduced a vision-based system that can interpret a user's hand gesture in real time to manipulate objects within a medical data visualization environment. Dinh et. al. [4] proposes a hand gesture classification system that able to efficiently recognize 24 basic signs of American Sign Language with Haar-like feature and AdaBoost learning algorithm.

In this paper, we proposed a system that can recognize the 12 important Thai alphabet letters selecting from the specific group using hand movement trajectory features. The paper is organized in six sections by the following: proposed hand movement recognition system, hand detection and tracking, trajectory feature extraction, experimentation results, and conclusion respectively.

II. Proposed System

In this section, we introduce the system for hand gesture recognition based on hand motion by analyzing the image sequences, obtained from the webcam attached over the computer screen. There are three main parts of our system: hand detection and tracking, hand feature extraction and gesture recognition. The overall of system is illustrated in the Fig. 2. First part, a hand detector is implemented using a cascade of boost classifiers, which allows obtaining very robust object detector. Two types of hand pose with vertical position are considered such as open and close hand, which is necessary for executing a command as start and stop symbol respectively. However, hand pose can have many figure caused by translation and rotation in 3D, which needs to be found. The tracking method, CamShift and Kalman filter, is then applied in order to extract hand through image sequence. In the second part, the gesture trajectory features are extracted and enhanced by the Douglas–Peucker algorithm which gives us a smallest number of control points trajectory. The features are selected to give a discrete vector that use as input to discrete HMM for recognizing commands in the final part.

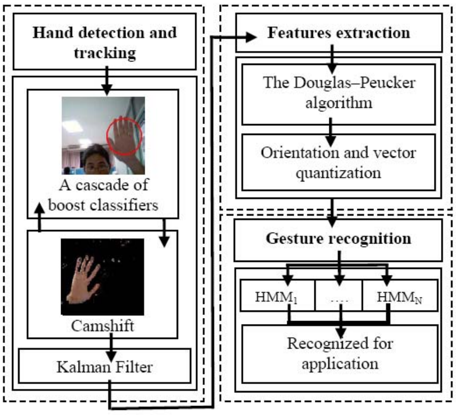

Hand detection and tracking
A cascade of boost classifiers
Camshift
Kalman Filter
Features extraction
The Douglas–Peucker algorithm
Orientation and vector quantization
Gesture recognition
HMM$_1$
....
HMM$_N$
Recognized for application

Fig. 1. System overview: (left) Hand detection and tracking. (right) feature extraction and recognition process.

Fig. 1 (left) shows the sequence of steps for hand detection and tracking, Fig. 1 (right) show the feature extraction and recognition process. These three parts will be detailed in the section 3, 4 and 5 respectively. In section 6, the experimentation result is discussed for evaluating the performance of system.

Manuscript received April 2, 2012; revised May 14, 2012.

The authors are with the Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University, P.O. Box 2 Kohong, Hatyai, Songkla 90112 Thailand (e-mail: jalego3@hotmail.com, kom@coe.psu.ac.th).

III. Hand Detection and Tracking

A. Hand Detection

A Cascade of Boost Classifier [5][6] for object detection is originally developed by Viola and Jones. This method uses Haar-like features and a cascade of boosted tree classifier as a supervised statistical model of object recognition. Haar-like feature consists of two or three connected "black" and "white" rectangles. The feature value is defined by the difference between the sums of pixel values within the black and white rectangles. Fig. 2 shows a basic set of Haar-like features.

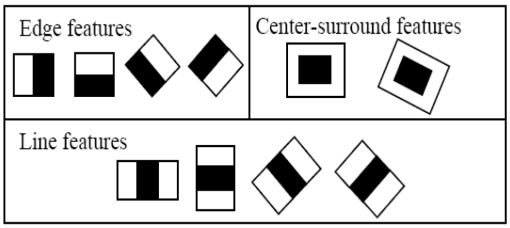


Fig. 2. A set of Haar-like feature.

The AdaBoost algorithm is introduced to improve the classification performance that designed to select rectangle feature which best separates the positive and negative example. In the first iteration, the algorithm trains a weak classifier *h(x)* using one Haar-like feature that achieves the best recognition performance for the training samples. The classifier consists of a feature *f(x)*, a threshold θ_t and parity p_t indicating the direction of inequality sign.

$$h_t(x) = \begin{cases} 1\square & p_t f_t \ x < p_t \theta_t \\ 0\square & \end{cases} \tag{1}$$

In the second iteration, the training samples that were misclassified by the first weak classifier receive higher weights. The iteration goes on and the final result is a cascade of linear combinations of the selected weak classifiers $H(x)$, which achieves the required accuracy.

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{2}$$

In the detection process by using the trained cascade, the sub-windows must be testing each stage of the cascade. A negative outcome at any point leads to the immediate rejection of the sub-window.

In our technique, a command begins with open hand posture and stop using close hand posture. Each frame, these two gestures must be detected. In the training processing, we have collected around 5,000 positive samples for each hand style from students in our university in various conditions such as indoor with neon light and outdoor with natural light. Systematically, around 10,000 negative samples are selected from landscape, building, and human faces or body images. Fig. 3 shows some of these samples. Each posture type was trained two rounds. The first round is for discriminating quickly the target posture from the outliers with small samples. By observing the false positive and false negative, we determine the additional positives and negative samples in order to overcome the better recognition rate while reducing noise as shown in Fig 4.

Fig. 3. Training samples: (left) positive samples. (right) negative samples.

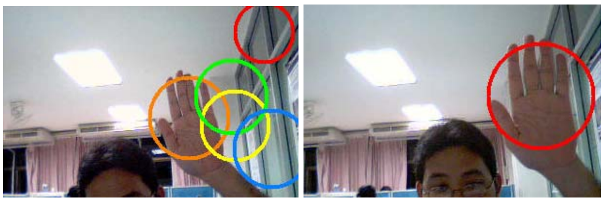


Fig. 4. Hand detection results: (left) 1st round. (right) 2nd round.

Table I show the detection rate results from our experimentation of the two hand postures with respective configurations. We notice that a good detection rate is raised at least more than 91% with 16 training stage at minimum.

TABLE I: Hand detection rate.

Hand posture			Training and Performances		
	Window size	Positive	Negative	Stage	Recognition rate
Open Close	32x32 32x32	5,058 5,678	10,448 10,448	19 16	91.17 % 99.06 %

B. Hand Tracking

CamShift [7] is a non-parametric technique using for color object tracking deriving from the Mean Shift algorithm. The main difference between CamShift and Mean Shift algorithm is that CamShift updates continuously its probability distributions; in generally the target object in image sequences changes significantly its shape size or color, while Mean Shift is based on static distributions. That why CamShift is suitable for tracking the rigid object. In the hand tracking, the process can be described as the following.

Step 1: the color probability distribution of detected hand image is determined from its histogram via hue component of HSV color space, related to skin color.

Step 2: this target distribution of detected hand is tracked on the searching window of next frame in image using mean shift algorithm. The mean shift vector, which is aimed for finding an optimized path that climbs the gradient of a probability distribution to the mode (peak) of nearest dominant peak, is necessary to be computed.

Step 3: the back-projection technique, which associates the pixel values in the image (tracking hand) with the value of the corresponding distribution, is applied.

Step 4: the center of mass and size of tracking hand (projected image) is computed and defined as hand features. This step will be detailed in the next section (hand features).

Step 5: on the next iterative, the current position of hand in image is used for defining the searching window on the next frame. The process is repeated at step 2 continuously.

Note that the step 1 will be re-executed systematically if the detected hand by Haar-like features with boost cascade of classifier found in the searching window. We found that Haar-like method provides very accurate results when hand is paralleled to the vertical axis, compared with CamShift, but missed mostly in other directions, so that CamShift is applied in order to solve the problem.

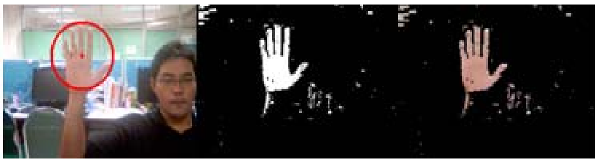


Fig. 5. Hand tracking with CamShift

C. Hand Movement Estimation

The Kalman filter [8] is a recursive linear filtering method. It addresses the general problem of trying to estimate the state of discrete time process that is described by the linear stochastic differential equation by the following.

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{3}$$

With a measurement $z \in \Re^n$ that is

$$z_k = Hx_k + v_k \tag{4}$$

The random variables w_k and v_k represent Gaussian noise of the process and measurement respectively. The algorithm of Kalman filter estimates a process by using feedback control technique: estimating the process state by an appropriate model and doing feedback by noisy measurements. As such, the equations of Kalman filter are formed into two groups: prediction and correction equations. In the post tracking of hand, the algorithm can be described as the following Fig 6.

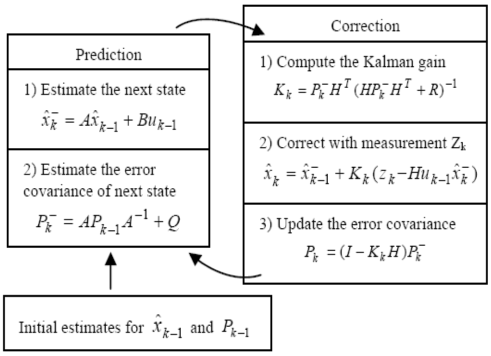

Prediction
1) Estimate the next state
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
2) Estimate the error covariance of next state
$P_k^- = AP_{k-1}A^{-1} + Q$
Correction
1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
2) Correct with measurement Zk
$\hat{x}_k = \hat{x}_{k-1}^- + K_k(z_k - Hu_{k-1}\hat{x}_k^-)$
3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Fig 6. Kalman filter process.

Experimentally, we found that hand detector by Haar-like features with boost cascade of classifier cannot provide total results and CamShift may wrongly track especially when there are other parts of body such as face or background, having color in skin color range, move close to tracking hand. Therefore, the Kalman filter is applied in order to predict hand position in the next frame based on previous frame, obtained by CamShift. To apply the Kalman algorithm, two principle equations needed to be declared: tracking process and measurement. The state of tracking process is measured from hand position and velocity in each image frame. So, we define the process of state x_k by the following:

$$\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k,i} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k-1,i} + w_{k-1,i} \tag{5}$$

where x, y, v_x, v_y are the position and the velocity of hand in the kth image frame respectively. Here, we assume that that the motion of hand between two successive frames can be uniformly approximated as straight line, the frame interval *Δt* is very short. For the measurement z_k, we define directly by the position value obtained from CamShift algorithm.

IV. Feature Extraction

A. Trajectory Parameter

In order to form the trajectory of hand position from the step 4 of hand tracking process by CamShift algorithm and Kalman filter. We choose to use the center point of hand and the trajectory can be obtained by joining this point in every frame in the sequence, in each frame, we obtained the center point of hand's region that can be easily computed from the moments of pixels in hand's region, which is defined as :

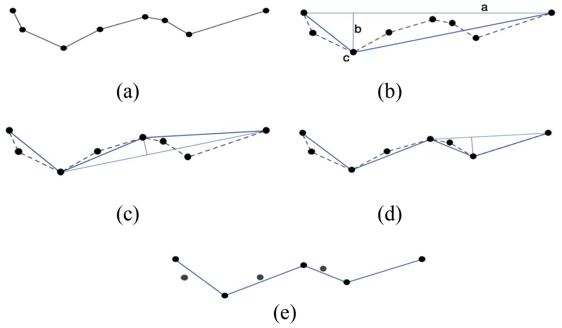
$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y) \tag{6}$$

In the above equations, *I(x,y)* is the pixel value at the position *(x,y)* of the image, *x* and *y* are range over the hand's region. The center point of hand *(X_c,Y_c)* is calculated as :

$$Xc = \frac{M10}{M00}, \quad Yc = \frac{M01}{M00} \tag{7}$$

B. Trajectory Approximation

Since during gesturing the hand does not move very fast, the position of hand does not change much from one frame to the next one. Therefore, for trajectory formation it may not be necessary to consider all the frames in a gesture sequence. Accordingly, we propose to select key trajectory point to be stored in the new sequence. This reduces the memory requirement as well as speeds up the trajectory matching during recognition process. The algorithm that we use to extract key trajectory point is the Douglas–Peucker algorithm [9]. The algorithm recursively divides the line. Initially it is given all the points between the first and last point (Fig 8. (a)). It marks the first and last point. It then finds the point that is furthest from the line segment with the first and last points as end points (this point is obviously furthest on the curve from the approximating line segment between the end points). If the point is closer than threshold to the line segment then any points not currently marked to keep can be discarded without the smoothed curve being worse than threshold. If the point furthest from the line segment is greater than threshold from the approximation then that point must be kept. The algorithm recursively calls itself with the first point and the selected point and then with the selected point and the last point (which includes marking the selected point being marked as kept).When the recursion is completed a new output curve can be generated.



(a) (b)

(c) (d)

(e)

Fig. 7. Smoothing a piecewise linear curve with the Douglas–Peucker algorithm.

A. Features

After getting the key trajectory points, we calculated the orientation between consecutive points to obtain a sequence of angle.

$$\theta_t = \arctan\left(\frac{Y_{t+1} - Y_t}{X_{t+1} - X_t}\right) \; ; t = 1,2,?T- \; 1 \qquad (8)$$

where T represents the length of gesture trajectory. The angle's domain is [0, 360] degrees. We divide this angle by 20^0 to quatize them to 18 directional codewords from 1 to 18. The codewords is used as input to HMM recognition model.

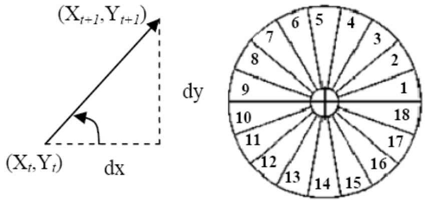


Fig. 8. 18 directional codewords

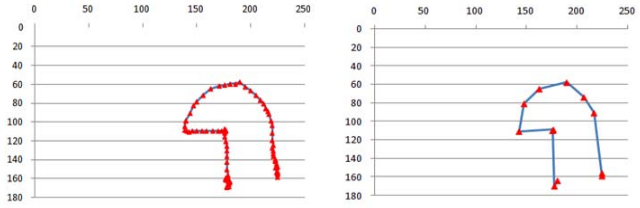


Fig. 9. Trajectory approximation: (left) Gesture curve (97 point); (right) Approximate gesture by Douglas-Peucker algorithm (11 point)

Fig. 9 shows the result of gesture that was approximate by the Douglas–Peucker algorithm. Table 2 define a gesture in Fig. 9 as a sequence of directional codewords O = { 13, 5, 3, 10, 5, 3, 1, 16, 16, 14, 5} which is used as input data for recognition process.

V. Recognition Model

A. Hidden Markov Model

In the recognition process, the probabilistic approach, such as the Hidden Markov model [10], is applied for characterizing gestures representing the Thai alphabet letters. The HMM model can be defined by the following:

1). The set of states $S = \{ s_1, s_2, ..., s_N \}$ where N is number of states.

2). An initial probability for each state π_i , $i = 1, 2, ..., N$ such that $\pi_i = P(s_i)$ at the initial step.

3). An N-by-N transition probability matrix, $A = \{a_{ij}\}$, where a_{ij} is the transition probability of taking the transition from state i to state j.

4). The set of observation symbols $O = \{o_1, o_2, ..., o_M\}$ representing our 18 directional codewords, M is the number of observation symbols.

5). An N-by-M observation matrix, $B = \{ b_j(o_k) \}$ where $b_j(o_k)$ give the probability of emitting observation symbol o_k from state j.

An HMM requires specification of two model parameters (N and M), specification of observation symbols, and the specification of the three probability measures: A, B, andπ. For convenience, we use the compact notation to indicate the complete parameter set of model.

$$\lambda = (A, B, \pi) \qquad (9)$$

There are three basic problems for HMM. These problems are the following:

Evaluation problem: Given the observation sequence O = $O_1O_2...O_T$, and model λ = (A, B, π), calculate the probability that model λ has generated sequence O.

Decoding problem: Given the observation sequence O = $O_1O_2...O_T$, and the model λ, calculate the most likely sequence of hidden states s_i that produced this observation sequence O.

Learning problem: How do we adjust the model parameters λ = (A, B, π) to maximize P(O|λ).

The solutions to these three problems are Forward-Backward algorithm, the Viterbi algorithm, and the Baum-Welch algorithm respectively.

TABLE II: Gesture Codeword.

Point	Degree	Codeword
(181,164)	-	-
(178,170)	243.43	13
(176,110)	91.90	5
(177,109)	45.00	3
(143,111)	183.36	10
(148,81)	80.53	5
(163,65)	46.84	3
(190.58)	14.53	1
(207,74)	316.73	16
(225,159)	276.70	14
(225,156)	90.00	5

VI. Experimentation Result

Our system is implemented using OpenCV library. Testing system is run on Intel processor Core 2 Duo, 2 Ghz, 2 GB memory. The video sequence is analyzed with image resolutions 320x240 pixels at 30 fps. The 12 Thai alphabets are chossen as commmand symbols, selecting from the 12 groups of 44 Thai alphabets [11] considering from its similarities, as shown in table III and Fig 10.

TABLE III: The 12 Groups of Thai Alphabet.

Group	Alphabet	Group	Alphabet	Group	Alphabet
1	ก ญ ฎ ฏ ถ ภ	5	ฑ ท ห	9	ผ ฝ พ ฟ ฬ
2	ข ฃ ช ซ	6	ฐ ธ ร	10	ฆ ม
3	ณ ฒ ฌ ว จ ง	7	ฉ น	11	ล ส
4	ค ด ต ศ	8	บ ป ย ษ	12	อ ฮ

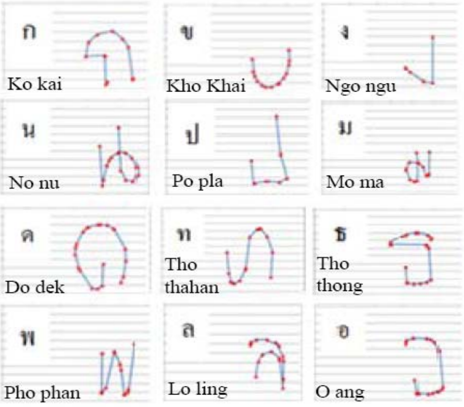


Fig. 10. Hand trajectory of 12 thai alphabets

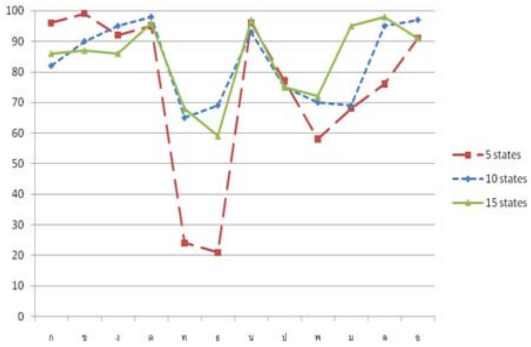


Fig. 11. Recognition rate for 12 Thai alphabets.

A. Thai Alphabet Recognition

In the evaluation, HMM topology is a fully connected with 3 types of states. About the number of state we take in our consideration the number of segment parts that are contained in possible gesture. The main direction part of all gesture in configuration consists of 8 directions (up, down, left, right, up-right, up-left, down-right and down-left). Intuitionally, we use 8 hidden states with 2 auxiliary states as initialization and output. However, the number of HMM cannot defined precisely by that consideration, it need to be tested with variation. Therefore, we also evaluate the HMM with 5 and 15 states for comparison with 10 states.

For each alphabet, we use 50 observation sequences for training and 100 observation sequences for testing. Table IV and Fig. 11 show the result, we can notice that the HMM with 10 states improvement significantly the result comparing to HMM with 5 state (9%). Although the HMM with 15 states give in global better result than HMM with 10 states but the improvement of correction rate is very insignificant (1%). In conclusion, the HMM with 10 states is good enough for our Thai alphabet recognition system. The demonstration of our system in real-time show in the video at http://www.youtube.com/watch?v=jEZi02EuweY

TABLE IV: Recognition rate of 12 Thai alphabets.

Gesture model	*Correct recognition*		
	5 state	*10 state*	*15 state*
1	96	82	86
2	99	90	87
3	92	95	86
4	95	98	96
5	24	65	68
6	21	69	59
7	96	93	97
8	77	75	75
9	58	70	72
10	68	69	95
11	76	95	98
12	91	97	91
μ	74.41	83.16	84.16
σ	27.40	12.81	12.79

VII. Conclusion

We have introduced the recognition system for the 12 Thai alphabets using hand trajectory features with Hidden Markov Model. The appropriate feature parameter and HMM topology were applied in our system. The result shows that the system can recognize in overall about 84 % of recognition rate. Additionally, more features may be introduced in order to improve the correction rate.

References

[1] W. Du and H. Li, "Vision based gesture recognition system with single camera," 5th International Conference o n ICSP, vol. 2, pp.1351-1357 vol.2, 2000.

[2] M. Elmezain and A. Hamadi, "Gesture Recognition for Alphabets from Hand Motion Trajectory Using Hidden Markov Models," *Signal Processing and Information Technology*, 2007 IEEE International Symposium on , vol, pp.1192-1197, 15-18 Dec. 2007

[3] J. Wachs, H. Stern, Y. Edan, M. Gillam, C. Feied, M. Smith, and J. Handler, "A Real-Time Hand Gesture System Based on Evolutionary Search," Vision. vol. 22, no. 3, Dearborn, Mich, Society of Manufacturing Engineers, 2006.

[4] Q. Chen. N. D. Georganas, and E. M. Petriu, "Real-time Vision-based Hand Gesture Recognition Using Haar-like Features," *Instrumentation and Measurement Technology Conference Proceedings*, 2007. IMTC 2007. IEEE, pp.1-6, 1-3, 2007.

[5] P. Viola, and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. of the IEEE CVPR*, vol. 1, pp. I-511- I-518, vol.1, 2001.

[6] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," *International Conference on Image Processing*, vol. 1, pp. I-900- I-903, 2002

[7] G. R. Bradski, “Computer vision face tracking for use in a perceptual user interface,” *Intel Technology Journal*, 1998.

[8] G. Welch and G. Bishop, “An Introduction to the Kalman Filter,” Annual Conference on Computer Graphics and Interactive Techniques. ACM Press, Addison-Wesley, Los Angeles, CA, USA (August 12–17), SIGGRAPH 2001 course pack edition.

[9] Ramer-Douglas-Peucker algorithm, July, 2010, [Online]. Available: http://en.wikipedia.org/wiki/Ramer-Douglas-Peucker_algorithm

[10] L. R. Rabiner, “A tutorial on hidden markov models and selected application in speech recognition,” in *Proc. IEEE 77*, pp 267-293, 1989.

[11] Thai script, october, 2010, [Online]. Availabl: http://en.wikipedia.org/wiki/Thai_script.

Kittasil Silanon was born in Narathiwat, Thailand, on April 14, 1986. He received the B.Eng (Computer Engineering) in 2008, from Prince of Songkla University (PSU), Hatyai,Songkla, Thailand.

He is currently pursuing M.Eng (Computer Engineering) at PSU in computer vision. His research interests are in the areas of image processing and computer vision.

Nikom Suvonvorn was born in Trang, Thailand, on November 26, 1976. In 2006, He received a PhD in computer science from l’Université de Paris Sud (XI), Orsay, France. In 2003, He obtained a DEA (Diplôme d’Etudes Approfondies), on Electronic System and Information Processing (SETI) from l’Institute d’Electronique Fondamontale (IEF) at the same university. In that year, He also got another master’s degree on computer engineering from Ecole Supérieure de Mécanique et d’Electricité (ESME)-Sudria engineering school, Paris.

He is currently a lecturer and research scientist at Department of Computer Engineering (CoE), Faculty of Engineering (ENG), Prince of Songkla University (PSU), Hatyai, THAILAND. His research corresponds to computer vision, image processing, and its related applications. The actual research is emphasized on the OpenVSS project, the next generation multimedia technologies applied for the Surveillance & Smart Environment System