Online Learning With Novelty Detection in Human-Guided Road Tracking

Jun Zhou, Li Cheng, and Walter F. Bischof

***Abstract*—Current image processing and pattern recognition algorithms are not robust enough to make automated remote sensing image interpretation feasible. For this reason, we need to develop image interpretation systems that rely on human guidance. In this paper, we tackle the problem of semiautomatic road tracking in aerial photos. We propose an online learning approach that naturally integrates inputs from human experts with computational algorithms to learn road tracking. Human inputs provide the online learner with training examples to generate road predictors. An ensemble of road predictors is learned incrementally and used to automatically track roads. When novel situations are encountered, control is returned back to the human expert to initialize a new training and tracking iteration. Our approach is computationally efficient, and it can rapidly adapt to dynamic situations where the image feature distributions change. Experimental results confirm that our approach is effective and superior to existing methods.**

***Index Terms*—Aerial images, human–computer interaction (HCI), image interpretation, novelty detection, online learning, road tracking.**

I. Introduction

ALTHOUGH image processing and pattern recognition methods for remote sensing are developing rapidly, there is still a huge gap between the requirements of most image interpretation applications and the accuracy and reliability achieved by computational methods. Many attempts to automate these tasks have been too fragile, and they require checking by experts before any final decision can be made. For this reason, many successful systems retain the "human in the loop," where a human operator can aid the automatic image interpretation through human–computer interactions (HCIs) [1]–[4].

Zhou *et al.* [5] proposed a general framework for human-guided image interpretation, which consists of the following five components: 1) a human–computer interface; 2) a user model; 3) computational algorithms; 4) a knowledge transfer scheme; and 5) a performance evaluation scheme. Among these five components, the interface and the computational algorithms have been extensively studied [6]–[9]. In contrast, research on the learning of computational algorithms from humans has been very limited.

Fig. 1. Image sample of size 663 × 423 pixels extracted from an orthorectified aerial photo. Notice that the occlusions, crossings, and the change in road surface materials lead to abrupt changes in road appearance.

In most existing semiautomatic systems, the role of the human operator is simply to provide initial parameters or to modify the image interpretation results, leaving the rest of the tasks to the predefined automatic model. Semiautomatic road extraction is such an example, which normally requires manual road seed input. The road seed information includes at least one of the following: starting point, edge location, road direction and width, as well as 2-D road templates. After an initial human input, the road extraction models perform either automatic road tracking using methods such as road edge following [10] and Kalman filtering [11], or automatic road optimizing using methods such as least square template matching [12]–[14], multiresolution modeling [15], [16], and active contours [16], [17]. In some cases, the interaction interface and the error correction tool were also designed to coordinate human actions with computer predictions [18], [19].

Image interpretation systems without learning have several problems. First, it is usually difficult to obtain a robust image interpretation model at the beginning of the system design. Thus, the computational model may fail frequently and require intensive human involvement, leading to a decrease in system efficiency and usability. More specifically, it is hard for the computational model to characterize the dynamic variations of the image features from multimodal and changing distributions. This is particularly the case in road tracking, where road features vary considerably due to changes in road materials, occlusions of the road, and lack of contrast with off-road areas. An example of a road scene is shown in Fig. 1, which illustrates

Manuscript received November 6, 2006; revised February 26, 2007. The work of W. F. Bischof was supported by a Discovery Grant from the National Science and Engineering Research Council of Canada. The work of J. Zhou was done when he was a Ph.D. candidate in the Department of Computing Science at University of Alberta, Edmonton, Alberta T6G 2E8, Canada.

J. Zhou and L. Cheng are with the Canberra Laboratory, National ICT Australia, Canberra, ACT 2612, Australia, and also with the Research School of Information Sciences and Engineering, Australian National University, Canberra, ACT 0200, Australia (e-mail: jun.zhou@nicta.com.au; li.cheng@nicta.com.au).

W. F. Bischof is with the Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8, Canada (e-mail: wfb@ualberta.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2007.900697

these dynamics. Second, there lacks the mechanism to enhance the computational model from multiple human inputs, thus the combination of human and computer resources remains suboptimal. To solve these problems, we need to bridge the gap between human inputs and automatic image interpretation using machine learning. This permits each human input to contribute to the building of a better image interpretation system.

In this paper, we propose an online learning approach that naturally integrates guidance from human experts with computational algorithms for tracking roads in aerial photographs. Human inputs provide the online learner with training examples to incrementally generate an ensemble of road profile predictors. The predictors are then used to automatically track roads. When novel road situations are encountered, control is returned back to the human expert. With this approach, we avoid the problem of having to explicitly define an off-road class, while enabling explicit learning from human inputs. Our learning algorithm is a kernel-based online learning approach for novelty detection, which is derived from one-class support vector machines (1-SVMs). The learned predictors are represented as weighted combinations of training examples, and the weights for each training example are derived from the large margin principle. We discuss the learning methods to acquire single and multiple predictors from one human input, and we show that they are equivalent to learning an expansion model or multiple drifting models that characterize human knowledge of the dynamics of the image patterns. We also introduce two tracking algorithms that use either optimal candidate–predictor combinations or multiple hypotheses. The experimental results on the learning and tracking models are compared and analyzed. The proposed approach is computationally efficient, and it can rapidly adapt to dynamic situations where the road features change. Experimental results confirm the effectiveness of our approach, and it is shown to be superior to existing methods.

II. Related Work

Interactive machine learning (IML) [20], [21] enables human operators to train predictors, e.g., decision trees or neural networks, by providing labeled training data and by defining boundaries between true and false instances. Such learning methods have been successfully used in image segmentation.

Muneesawang and Guan [22] employed IML in a model for content-based image retrieval. An initial image distance function is given using a radial basis function (RBF) network. When a user supplies a query image, the system retrieves images in the databases that are closest to the query image. Then, the user indicates correct and incorrect retrievals. The image features are used to train the RBF predictor using learning vector quantization method, so that the system can retrieve a new set of images. This process is repeated until the user is satisfied with the retrieval results.

Maloof *et al.* [23] proposed a similar model, which enables interactive supervised learning of building detection in aerial images. In each training session, the classifiers generate roof candidates. The human analyst provides positive and negative labels for these candidates, which in turn are used to update the classifiers. The training session terminates when all training samples have been classified. Four types of classifiers were implemented, namely: 1) a nearest neighbor method; 2) a naive Bayesian classifier; 3) decision trees; and 4) a continuous perceptron. In the testing session, these classifiers were used to automatically detect buildings in testing data sets.

The aforementioned methods acquire a single predictor from human inputs. It is also possible to acquire a set of predictors if multiple predictors are generated from human inputs. In this case, the problem can be considered as predicting from experts' advice [24]. Littlestone and Warmuth were among the first to tackle this problem [25]. They proposed a weighted majority algorithm that can be summarized as follows.

1) Set the initial weight w_i for each expert i to 1, where $1 \leq i \leq n$, and n is the number of experts.
2) The algorithm compares the total weight from experts that predict output 0 and the total weight from experts that predict output 1, and outputs the prediction of the larger total.
3) The weights of the experts that make mistakes are multiplied by a penalty factor between 0 and 1.

Kolter and Maloof [26] extended the aforementioned algorithm to create and delete experts in concept drift problems using a weighted majority algorithm. In addition to the predictions by each expert, a global prediction is calculated as the label with the highest accumulated weight. Whenever the global prediction is incorrect, the algorithm creates a new expert. To constrain the number of experts, the algorithm eliminates experts with weights less than a certain threshold.

As described in a later section, our online learning problem is similar to the concept drifting problem. Each predictor is a weighted sum of past training examples, where the weight for each training example is derived from the large margin principle [27]. Instead of creating new predictors automatically, we maintain a dynamic list of predictors. One set of predictors can be generated from one iteration, and different iterations generate different predictor sets.

We developed our algorithm for novelty detection, i.e., for identifying new or unknown data that were not present in the training stage [28]. Thus, it is important to find true novel samples in testing while minimizing false positives. Cheng *et al.* [29] proposed an online learning method with sparse kernels to solve learning problems with examples drawn from a nonstationary distribution. We extend this method to online learning from human inputs for novelty detection, so that it can fit into the HCI framework for image interpretation applications.

III. System Framework

In this section, we give an overview of the main procedures of the proposed approach on human-guided road tracking. A road tracking task starts with an aerial photograph containing a number of roads within which only one is the target of tracking. The system proceeds with iterations consisting of human input, sampling, and prediction phases. The first phase is aimed at learning from human input, whereas the last phases are aimed at automatic road tracking.

A. Input Phase

A human input consists of two mouse clicks on an image, with the line joining the click positions defining a road axis and indicating the road direction. Along the road direction, a set of 2-D road profiles, which are represented as vectors of image gray levels, are extracted normal to and along the road axis at consecutive axis points. All vectors have the same length d, which is the sum of the length of the two 1-D profiles. The length of the 1-D profiles is estimated from the road width, which is calculated as the distance between the road edges. This, in turn, is obtained with a gradient-based edge detection method [30]. We denote a 2-D road profile as $x \in \mathcal{X} \subseteq \mathbb{R}^d$, where $\mathcal{X}$ is the profile space with dimension d. We denote one human input as one learning session $s \in \mathcal{I}_S$, where $\mathcal{I}_S$ indexes the set of learning sessions that occurred during the road tracking task. A road profile x is associated with a label $y \in \mathcal{Y}$, where $\mathcal{Y}$ is defined as

$$\begin{cases} y = 1, & \text{on the road} \\ y = 0, & \text{off road} \end{cases} \tag{1}$$

and a state $\sigma \in \Sigma$, which encodes a location, as

$$\sigma = [\, u \quad v \quad \theta \,]' \tag{2}$$

where u and v are the coordinates of the road axis point, and θ is the direction of the road. The triplet $z = (x, \sigma, y)$ represents a training example extracted from human input. Later, during the automatic tracking step, observed candidate examples also take this form. All training examples are extracted from road axis points entered by the human, hence $y = 1$ for all training examples. A road profile predictor $f_S \in \mathcal{F}$ is learned from the set of training profiles in a learning session s, where $\mathcal{F}$ denotes the set of road profile predictors learned in sessions indexed by $\mathcal{I}_S$.

B. Sampling Phase

Automatic road tracking starts after each training phase. Suppose we have already found the road axis point at time $t-1$, and we would like to find the next road axis point at time t. Further, suppose the system has experienced a sequence of learning sessions $(1, \ldots, s)$ and obtained an ensemble of predictors $\mathcal{F}_1^s \triangleq (f_1, \ldots, f_S)$. It then proceeds with the sampling phase, where the system searches the neighborhood along the current road axis for candidate road profiles $\tilde{\mathcal{X}} = \{x_1, \ldots, x_m\}$, where m denotes the number of candidates being sampled. The candidates at time t are sampled at locations computed by the following nonlinear function:

$$\sigma_t = \begin{bmatrix} u_{t-1} + \varrho \cos(\theta_{t-1} + \varphi) \\ v_{t-1} + \varrho \sin(\theta_{t-1} + \varphi) \\ \theta_{t-1} + \varphi \end{bmatrix} \tag{3}$$

where $\varphi \in [-\pi/18, \pi/18]$ and ϱ (initially bounded in $[-3, 3]$) are angle and depth parameters, with different angles and depths generating different sampling locations.

C. Prediction Phase

After the sampling phase, we obtain a set of candidate profiles $\tilde{\mathcal{X}}$ and their associated locations. The goal of the prediction phase is to predict the road axis point at time step t and to decide whether the tracking is on the road, using the trained road profile predictors. As will be introduced later, there are two strategies for this purpose. The first strategy is to select an optimal candidate profile $\hat{x}_t$ and predict its label $\hat{y}_t$. In this case, we assume that, at an appropriate road axis location, there has to be exactly one profile with $y = 1$, that is, only one candidate profile can be the on-the-road profile. The second strategy is to select a set of supporting candidate profiles and compute the location of road axis point as a weighted sum of profile locations. In this case, we assume that several road profiles can be on-the-road profiles.

If the prediction indicates that the tracking is on the road, the automatic tracking continues and returns to the sampling phase. Otherwise, a novelty has been detected and control is handed back to the human expert for further input. In this manner, switching between human inputs and automatic tracking continues until the tracking task is completed.

For predicting a candidate example $\hat{z}_t = (\hat{x}_t, \hat{\sigma}_t, \hat{y}_t)$, we hope to be close to the true one $z_t = (x_t, \sigma_t, y_t)$, that is, $\hat{x}_t \to x_t$, $\hat{\sigma}_t \to \sigma_t$, and $\hat{y}_t = y_t = 1$. Unfortunately, this is not always the case. None of the candidate profiles may be labeled correctly when occlusions, such as vehicles and shadows, are present on the road, or when radiometric road properties change, for example, when encountering a road segment built with different materials. A heuristic strategy is developed to overcome these often-encountered situations and to improve the tracking efficiency. We employ a jump-over strategy, where the upper and lower bounds of the step size ϱ in (3) is increased to jump over the current location when the system fails to find a road profile with $\hat{y} = 1$. Then, a new sampling phase occurs. When failures continue, even with the jump-over strategy, the system recognizes a tracking failure and returns control to the human expert, who then inputs another road segment from which new training examples with $y = 1$ can be extracted.

In summary, the proposed approach is naturally decomposed into the following two steps: 1) a learning algorithm devised for the human input phase and 2) a tracking algorithm for automatic road tracking.

IV. Learning Algorithms

The interactions between human and computer lead to a situation where learning sessions are mixed with automatic tracking runs. The first learning session is initialized by the first human input. Each successive learning session starts when a road outlier is detected for the current prediction and control is handed back to the human expert.

A. Basic Learning Algorithm

For the sake of simplicity in describing the algorithm, we assume that each learning session contains exactly S training

examples $(z_{t+1}, \ldots, z_{t+S})$, where $z_i = (x_i, \sigma_i, y_i), \forall i \in \{t+1, \ldots, t+S\}$.

The basic learning algorithm aims at obtaining a reasonable road profile predictor f_s in one learning session. Given the current learning session s starts at time $t+1$, we define a kernel mapping $k(\cdot,\cdot)$ from profile space to a Hilbert feature space, $\mathcal{X} \to \mathcal{H}$ as $x \mapsto k(x,\cdot) \in \mathcal{H}$. Here, $\mathcal{H}$ denotes the reproducing kernel Hilbert space with induced kernel $k(\cdot,\cdot)$ such that $f(x) = \langle k(x,\cdot), f(\cdot)\rangle$, and $\langle\cdot,\cdot\rangle$ gives the inner product. The norm in this case is naturally defined as $\|\cdot\| = \langle\cdot,\cdot\rangle^{1/2}$.

As in the online learning algorithm described in [29], the road profile predictor $f \in \mathcal{H}$ can be represented as a weighted combination of training profiles, where past profiles in the learning session $\{x_i\}_{i=t+1}^{t+S}$ are associated with different weights $\{\alpha_i\}_{i=t+1}^{t+S}$ that are derived formally from the large margin principle [27]. In this paper, we extend this algorithm to incorporate learning from human inputs and to deal with the novelty detection scenario, so that the learning problem is naturally formulated as a novelty detection by solving online 1-SVMs.

Given a profile x_i at time i, the novelty detection can be formulated as an optimization problem, i.e.,

$$\min_f C \cdot (\rho - \langle f, k(x_i,\cdot)\rangle)_+$$

where $(\cdot)_+ \triangleq \max\{\cdot, 0\}$, C is a positive constant cutoff value for the penalty imposed on point prediction violations, $\rho > 0$ is the margin parameter, and f is bounded. This can be rewritten in a form similar to 1-SVMs [31], i.e.,

$$\min_{f,\xi} \frac{\lambda}{2}\|f\|^2 + C\xi \quad \text{s.t. } \langle f, k(x_i,\cdot)\rangle \geq \rho - \xi, \qquad \xi \geq 0 \tag{4}$$

where $\lambda > 0$ is a regularization parameter, and ξ is a slack variable.

According to the framework in [29], we want to minimize the risk function $R_{\text{div}}(f) + R_{\text{reg}}(f)$, where $R_{\text{div}}(f) = \|f - f_i\|^2/2$ measures the divergence of the predicted f from the previous prediction f_i. The second term is the Lagrangian function of the optimization problem in (4), i.e.,

$$R_{\text{reg}}(f) = \underbrace{\frac{\lambda}{2}\|f\|^2}_{\lambda R_{\text{cap}}(f)} + \underbrace{C\xi + \varsigma\left(\rho - \langle f, k(x_i,\cdot)\rangle - \xi\right) - \zeta\xi}_{R_{\text{inst}}(f)} \tag{5}$$

where ς and ζ are Lagrangian multipliers. Equation (5) consists of the following two terms: 1) the capacity risk $R_{\text{cap}}(f)$, which controls the complexity of the predictor f, and 2) the instantaneous risk $R_{\text{inst}}(f)$. We further introduce $\tau > 0$, the decay rate parameter, and an auxiliary variable $\upsilon > 0$, such that $\lambda = \tau/(1-\tau)$ and $\varsigma = \upsilon/(1-\tau)$.

We also define the magnitude of violation as

$$l_i \triangleq l(f_i; x_i) = \left(\rho - (1-\tau)\langle f_i, k(x_i,\cdot)\rangle\right)_+. \tag{6}$$

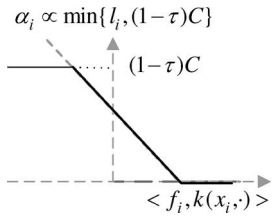


Fig. 2. Robust weight assignment α_i at time i. The horizontal axis is $\langle f_i, k(x_i,\cdot)\rangle$. The vertical axis is α_i and is upper bounded by $(1-\tau)C$.

TABLE I
LEARNING ONE ROAD PROFILE PREDICTOR FROM ONE HUMAN INPUT

Input: The cut-off value C, decay rate τ, margin ρ, current learning session s
Initialize: $f_s \Leftarrow \underline{0}$.
  for $i = t+1$ to $t+S$ **do**
    Observe profile x_i
    Compute $l_i^{(s)}$ according to Equation (6)
    Compute $(\alpha_j^{(s)})_{j=t+1}^{i}$ according to Equation (7)
  end for
Output: The sequences $(\alpha_j^{(s)})_{j=t+1}^{t+S}$, s, f_s.

By solving the (previously mentioned) risk minimization problem (see [29]), the separating function f turns out to be $f(\cdot) = \sum_{i=t+1}^{t+S} \alpha_i k(x_i,\cdot)$, where the weights are

$$\begin{cases} \alpha_j & \Leftarrow (1-\tau)\alpha_j \ \forall j < i \\ \alpha_i & \Leftarrow \min\left\{\frac{l_i}{k(x_i,x_i)},\ (1-\tau)C\right\}. \end{cases} \tag{7}$$

As stated in [29], the resultant weight updating formula has several advantages. First, α_i is always upper bounded by the constant $(1-\tau)C$. This ensures limited influence from outliers (see Fig. 2). Second, by adjusting the decay rate $\tau \in [0,1)$ to an appropriate value, the new predictor f is able to balance between two extreme situations, either fully adapting to the current example (i.e., forgetting all the past examples as $\tau \to 1$) or keeping all past examples in memory (i.e., becoming a batch learning case as $\tau \to 0$). Finally, at time i, the weight of current example x_i, i.e., α_i, is automatically obtained, while the weights for previous examples $\forall j < i$ are retained with proper decay, instead of being recomputed from scratch. Thus, this update formula allows for a rather efficient computation, as shown in Table I.

Based on the learning algorithm, the road profile predictor f_s is obtained given the weight sequence $\left(\alpha_i^{(s)}\right)_{i=t+1}^{t+S}$ and the corresponding training examples. For a sequence of length S, the space complexity of the proposed learning algorithm is $(d+1)S$ (where d is the dimensionality of the input road profile), and the time complexity is $O(S^2)$.

B. Learning Multiple Predictors From One Training Session

We can generate multiple predictors from one training session. The separating function is in the form of $f(\cdot) = \sum_{i=t+1}^{t+S} \alpha_i k(x_i,\cdot)$, hence given an input that consists successive examples $(z_{t+1}, \ldots, z_{t+S})$, we can compute up to S separating functions from $t+1$ to $t+S$. Thus,

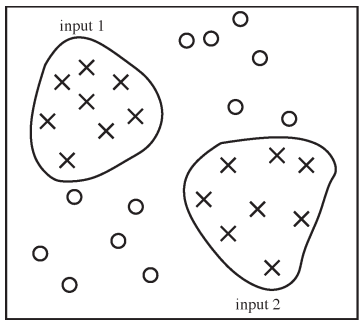


Fig. 3. Graph showing a set of input data. Initially, the class of the data is unknown. A human input sequentially marks the data that the human operator considers as positive, which are in turn used in one training session. The region delimited by one curve identifies these positive examples. In the proposed interactive model, multiple inputs can be observed and, thus, form multiple regions as marked by inputs 1 and 2.

an ensemble of predictors is given in the form of $\mathcal{F}_1^s \triangleq (f_{1,1}, f_{1,2}, \ldots, f_{i,j}, \ldots, f_{s,S-1}, f_{s,S})$. For a training sequence of length S, the time complexity of this extended model is not different from the single-predictor model, but the space complexity increases to $(d+1)S^2$.

In each training session, some predictors may not be as robust as others. To evaluate their robustness, we calculate the training errors for all predictors and rank them according to their performance. This permits us to select more robust predictors in the tracking runs and to achieve faster and more accurate results.

C. Discussion of the Learning Algorithms

As aforementioned, the online learning algorithm is derived from 1-SVMs [31], and it is thus driven primarily by positive training examples. This formulation perfectly fits the road tracking scenario, since the human inputs provide only positive examples for the learning process. In the following, we analyze the effect of applying this algorithm to a set of 2-D data (see Fig. 3). We assume a Gaussian kernel, and the learning algorithm generates, in each learning session, a sphere enclosing the input data features.

We consider two learning models. In one model, we use a decay rate $\tau = 0$, where all past examples contribute equally to a predictor without loss of any information. Learning thus expands the region to cover the new example. In the model with decay rate $\tau \neq 0$, the region drifts to adapt to the new example, and the extent of the drifting depends on the value of τ. With a larger τ, drifting is larger and past examples are forgotten more quickly (see Fig. 4).

Since old examples characterize situations that are considered positive, we may want to preserve the past training examples in the drifting model. In this case, multiple predictors can be learned from one human input. The predictors acquired in each time step are preserved, each becoming an independent expert for the tracking process. This model is shown in Fig. 5.

V. Tracking and Novelty Detection Algorithms

The tracking process automatically interprets road features in the aerial images using the trained predictors. If a novelty is found in the direction of the prediction, then either a new road condition has been encountered or a tracking failure has occurred. In both cases, human guidance is required to initialize a new training session, and if necessary, to correct the tracking error. We now introduce two tracking algorithms: one based on a single optimal predictor, and the other using multiple predictors.

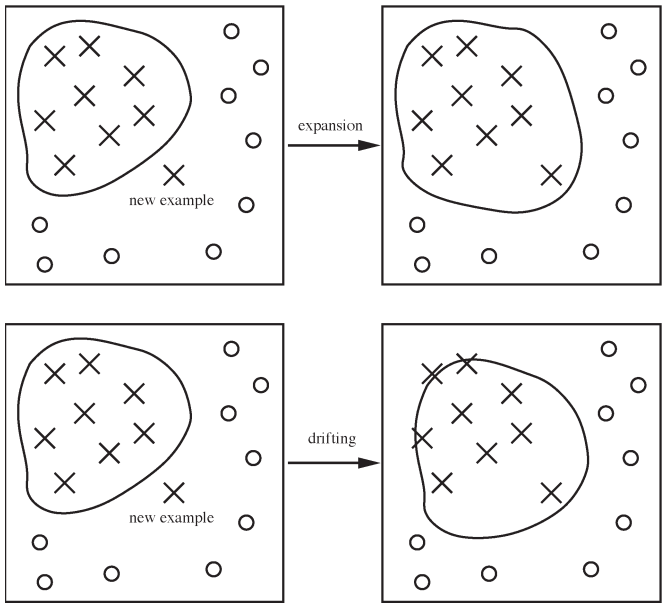


Fig. 4. Two models of learning, (upper graph) expansion and (lower graph) drifting. The left graphs shows the learning result up to time t. At time $t+1$, a new positive example is observed. The expansion model enlarges the region delimited by a curve to include the new example, whereas the drifting model moves the region to adapt to the new data.

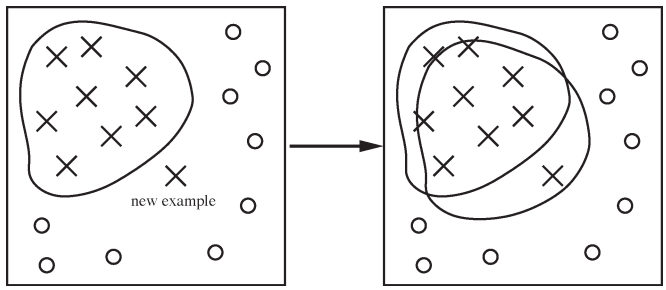


Fig. 5. Learning multiple predictors from one human input. The learned model at time t is shown in the left graph. At time $t+1$, a new model is learned to adapted to the new training example, while the old model is kept.

A. Optimal Candidate–Predictor Combination

We assume that, at time t, after the sth learning session, the newly learned road profile predictor f_S is incorporated into the set of predictors as $\mathcal{F}_1^s = \mathcal{F}_1^{s-1} \cup f_S$. Tracking starts by searching the neighborhood along the current road axis for candidate profiles. The violations of the candidates are calculated using (6), and the one with the minimum violation, i.e., $\hat{x}_t$, is picked as the input to the predictors. If it is considered to be on the road ($\hat{y}_t = 1$ with a predictor $f \in \mathcal{F}$, which produces the least violation), the state $\hat{\sigma}_t$ of the profile is used to set the current road axis point, the current road direction, as well as the origin of the next prediction. The tracking algorithm

TABLE II
TRACKING ALGORITHM I: OPTIMAL CANDIDATE–PREDICTOR COMBINATION

Input: Decay rate τ, margin ρ, threshold ϵ, the set of learned predictors so far $\mathcal{F}_1^s$.
Obtain a set of m candidate profiles $\tilde{\mathcal{X}}$ from the sampling phase.
for $i = 1$ to m, $j = 1$ to s **do**
Compute $l_{i,j}$ according to Equation (6)
end for
$(l^*, \hat{x}_t) \leftarrow \min_{i,j}\{l_{i,j}\}$
Predict label as

$$\hat{y}_t = \begin{cases} 0, & l^* \geq \epsilon \\ 1, & otherwise \end{cases}$$

Output: $\hat{z}_t = (\hat{x}_t, \hat{\sigma}_t, \hat{y}_t)$, s, f_s.

based on the optimal candidate–predictor combination is shown in Table II.

The tracking algorithm picks the candidate–predictor combination with minimum violation and evaluates whether the candidate is on the road and its reliability by comparing the violation value to a threshold ϵ.

B. Multiple-Hypotheses Support

The road axis point and the next prediction depend exclusively on the location and direction of the optimal candidate. Noisy candidates can thus mislead the prediction. To solve this problem, we propose a multiple-hypotheses method. The violations of the candidates are again calculated using (6) for each $f \in \mathcal{F}$ and are stored in a lookup table H whose entries $h_{i,j} \in \mathsf{H}$ correspond to hypothesized road axis points from candidate–predictor combinations.

Given the lookup table, we search for hypotheses whose violations are lower than a threshold ϵ. If a violation $l_{i,j}$ is smaller than ϵ, the hypothesis is a supporting hypothesis and its weight is set to

$$w_{i,j} = \exp(-l_{i,j}). \tag{9}$$

If $l_{i,j} \geq \epsilon$, the hypothesis is discarded. Finally, we calculate the road axis point as a normalized weighted sum of the supporting hypotheses. If the number of supporting hypotheses is zero, a novelty has been detected. The resulting tracking algorithm is shown in Table III.

TABLE III
TRACKING ALGORITHM II: MULTIPLE-HYPOTHESES SUPPORT

Input: Decay rate τ, margin ρ, threshold ϵ, the set of learned predictors so far $\mathcal{F}_1^s$, an empty H.
Obtain a set of m candidate profiles $\tilde{\mathcal{X}}$ from the sampling phase.
for $i = 1$ to m, $j = 1$ to s **do**
Compute $l_{i,j}$ according to Equation (6)
if $l_{i,j} < \epsilon$ **then**
compute $w_{i,j}$ according to Equation (9)
$h_{i,j} \leftarrow l_{i,j}$
$\mathsf{H} = \mathsf{H} \cup h_{i,j}$
end if
end for
if $\mathsf{H} \neq \varnothing$
$w_{i,j} = w_{i,j} / \sum_i \sum_j w_{i,j}$
$\hat{x}_t \leftarrow \sum_i \sum_j w_{i,j} x_i$
$\hat{\sigma}_t \leftarrow \sum_i \sum_j w_{i,j} \sigma_i$
end if
Predict label as

$$\hat{y}_t = \begin{cases} 0, & \mathsf{H} = \varnothing \\ 1, & otherwise \end{cases}$$

Output: $\hat{z}_t = (\hat{x}_t, \hat{\sigma}_t, \hat{y}_t)$

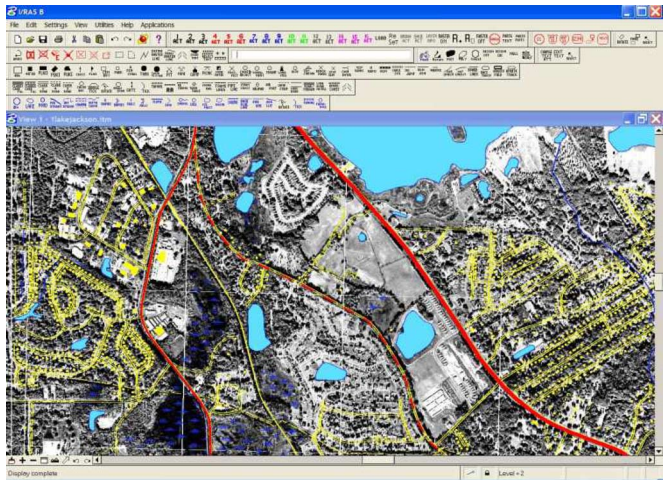


Fig. 6. Map revision environment. Old map layers are aligned with the latest digital image data so that the human operator can make visual comparison of the two resources.

VI. EXPERIMENTAL RESULTS

A. System Implementation

We have developed a semiautomatic road tracking system for the U.S. Geological Survey (USGS) map revision platform [32]. This platform is used to revise the USGS 7.5-min quadrangle topographic maps [33]. This map series consists of 55 000 map sheets in raster form (USGS also produces the digital line graphs, a vector form product, which also comprises a transportation layer). The revision of this map series is the Raster Graphic Revision program, which uses existing map sheets as the primary input and creates new maps as the primary output. The maps are displayed on a computer screen, together with the digital orthophoto quads (DOQs) of the area to be mapped. DOQs are orthogonally rectified images produced from aerial photographs taken at a height of 20 000 ft, with an approximate scale of 1 : 40 000 and a ground resolution of 1 m. A cartographer then makes a visual comparison of the map and the DOQ. When a discrepancy is found between a feature on the map and the DOQ, the cartographer modifies the map to match the DOQ. Fig. 6 shows the environment of this map revision platform.

Within this system platform, we have implemented an embedded software to keep track of the time, the type, and the location of each human input. The inputs related to road tracking were used for the experiments reported in Section VI-B.

B. Experimental Results

We conducted experiments with humans tracking road features. Eight users were involved in the manual road

TABLE IV
STATISTICS ON HUMAN INPUT

	user1	user2	user3	user4	user5	user6	user7	user8
total number of inputs	510	415	419	849	419	583	492	484
total time (in seconds)	2765	2784	1050	2481	1558	1966	1576	1552
average number of inputs per task	18.2	15.2	15.0	30.3	15.0	20.8	17.6	17.3
average time per task (in seconds)	98.8	99.4	37.5	88.6	55.6	70.2	56.3	55.4
average time per click μ	5.4	6.7	2.5	2.9	3.7	3.4	3.2	3.2

interpretation. These users were students who did not have experience in road interpretation. Before performing the actual annotation, each user was given 30 min to learn the road interpretation process as well as the software environment. They did so by working on a real aerial photograph that was different from the photograph used in the experiment. Because road interpretation is not a difficult task, all users became familiar with the annotation operations in less than 15 min. Then, they were assigned 28 tasks to annotate roads on the aerial photograph (one photograph of $14\,576 \times 12\,749$ pixels) of the Marietta area in Florida. In each task, one road had to be annotated. The tasks included a variety of scenes with transnational highways, intrastate highways, and roads for local transportation. Further, these tasks contained different road types and various road conditions. We obtained eight data sets, each containing 28 sequences of road axis coordinates marked by users. Table IV shows some statistics on the human data.

We simulated the semiautomatic road tracking process using the recorded human data as virtual users. These data were used to initialize the online learning, to regain control when the automatic road tracker failed, and to correct tracking errors. Finally, we compared the performance between the simulated semiautomatic road tracking and the complete manual tracking for each user.

Each human input initiated a new training session and generated a new road profile predictor. When multiple human inputs were encountered in a road tracking task, an ensemble of road profile predictor was generated to form the automatic road tracker, leading to an incremental improvement of prediction results. Notice that the length of a training sequence (S in Table I) in a learning session is only dependent on the length of the corresponding road segment entered by the human. In practice, when S is very large, the training example queue is truncated and only the latest 50 examples are kept to generate the road profile predictor.

For each road tracking task, a new road tracker is learned. This is natural because automatic road tracking is initialized by a manual road seed input, which leads to a learning session. The newly learned tracker can better adapt to the current road condition.

In all experiments, we used Gaussian kernels for the proposed algorithms. The standard deviation for the Gaussian kernel was set to $30\sqrt{2}$. Further, the decay factor τ was fixed to an appropriate constant even though it is a decreasing function of t in the theoretical analysis. The value of τ and its influence on tracking performance are reported in Section VI-C.

Some of the road tracking results are shown in Figs. 7–12. Fig. 7 illustrates an example of interactive road tracking. When a novelty was detected by the road tracker, it required human input to initialize the next tracking iteration. Fig. 8 illustrates how a trained road tracker deals with clear and shadowed road conditions. These road conditions had been experienced in the previous tracking process, during which corresponding road profile predictors had been generated from human inputs. Using the optimal candidate–predictor combination tracking strategy, an optimal predictor was selected to correctly predict the label of the road profiles in these conditions. Fig. 9 shows how the road tracker handled a junction using the same tracking strategy. When there are two options on the tracking direction, the tracker chose the direction with an optimal road profile. In some cases, the tracking may fail when facing a junction. Then, a human should get involved to guide the tracker. Fig. 10 illustrates how the road tracker deals with occlusions on the road. The tracker uses a jump-over strategy as introduced in Section III-C. Locations with occlusions are jumped over by increasing the step size of the prediction, if these occlusions

Fig. 7. Example of road tracking. Road tracking starts from the upper left area of the image. White line segments show the locations of human inputs; white dots are the detected road axis points. When the road changes from light to dark, a human input is required to guide the tracking because the dark road has not been experienced by the road tracker.

Fig. 8. Example of road tracking. Road tracking proceeds from the upper right to the lower left corner of the image. Black dots are automatically detected road axis points. The tracking is robust when facing occlusions on the road.

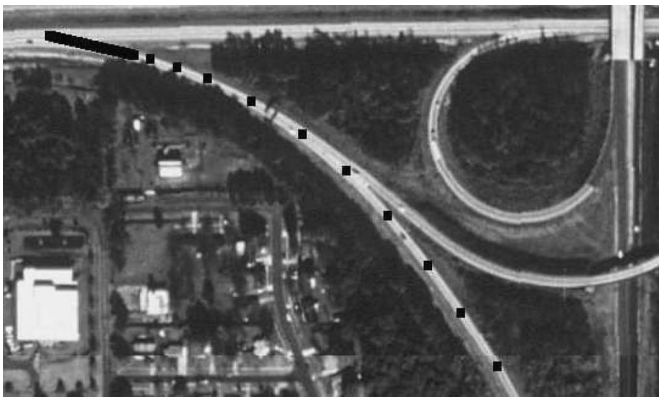


Fig. 9. Example of handling a junction. Road tracking starts from the upper left corner of the image. The black line segment is a human input, and black dots are the tracking result.

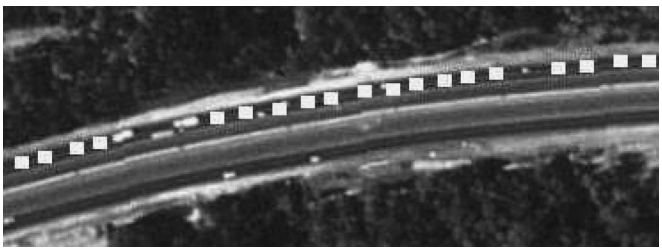


Fig. 10. Example of handling occlusions on the road. Road tracking proceeds from the lower left to the upper right corner of the image. The locations without accepted observation are jumped over.

Fig. 11. Example of accurate and inaccurate tracking on the same road. White line segments are the inputs from two users, and white dots are the detected road axis points. In the left image, the input is not at the center of the road. It generates a biased road profile predictor, which in turn makes inaccurate road tracking that is deviated from the true road centerline. The right image shows an accurate road tracking initialized by an accurate human input.

have not been experienced during previous human inputs. In Fig. 11, two different human inputs generated different tracking results on the same road. The quality of the human input greatly affected the tracking accuracy. Fig. 12 shows an inefficient

Fig. 12. Example of low efficient road tracking caused by complex road condition. The road tracking starts from the lower left corner of the image. White line segments are the human input, and white dots are the tracking result. Due to the complexity of the road, intensive human input is required.

tracking example. There are too many variations in road conditions, which cause frequent tracking failures and intensive human involvements.

To qualitatively evaluate the performance of the proposed algorithms, we followed the four criteria reported in [30]. These criteria evaluate the efficiency and the accuracy of the algorithms. Efficiency is defined by the savings in the number of human inputs, the savings in human tracking distance, and the savings in tracking time. The number of human inputs and tracking time are related to each other so that reducing the number of human inputs also decreases tracking time. Given an average time for a human input, we obtain the following empirical function to calculate the time cost of the road tracker:

$$tc_T = ta_T + \mu s \tag{11}$$

where tc_T is the total time cost to track the entire road up to road profile T, ta_T is the time for automatic tracking process, s is the number of human inputs required during the tracking, and μ is a user-specific variable, which is calculated as the average time for an input, i.e.,

$$\mu_i = \frac{\text{total time for user } i}{\text{total number of inputs for user } i}, \qquad 1 \le i \le 8. \tag{12}$$

The last row of Table IV shows, for each user, the average time for an input. Finally, tracking accuracy is defined as the root-mean-square error between the semiautomatic system and complete manual inputs.

We compared the results of the proposed online novelty detection (OND) algorithm with the two algorithms reported in [34]. Those two algorithms fit into the same HCI framework introduced in this paper. However, instead of using online learning from human inputs, those algorithms built a dynamic knowledge base to store the reference road profiles extracted from the human inputs. In the tracking process, observed profiles were matched with reference profiles using cross correlation. The state prediction was implemented with cross correlation Kalman filtering (CCKF) and cross correlation particle filtering (CCPF). Due to the factored sampling involved in the particle filtering, the tracker may perform

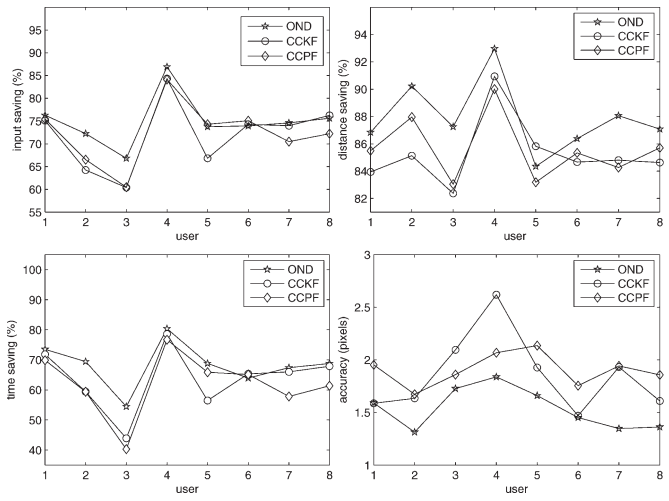


Fig. 13. Comparison of the tracking performance for the proposed algorithm (OND) and the cross correlation with Bayesian filtering algorithms (CCKF and CCPF). The horizontal axis shows data sets recorded from different users. The vertical axis shows different evaluation criteria. The parameters for the OND model are given as follows: $\tau = 0$, $\rho = 1$, and $\epsilon = 0.95$. For display purposes, the data points are connected by lines even though there is no quantitative relationship between data sets.

TABLE V
COMPARISON OF ROAD TRACKING RESULTS. ALGORITHMS AND COMPARISON CRITERIA ARE DESCRIBED IN THE TEXT. AVERAGE VALUES AND STANDARD ERRORS FOR EACH CRITERION ARE SHOWN

	input saving (%)	distance saving (%)	time saving (%)	RMSE (pixels)
CCKF	71.9 ± 2.7	85.3 ± 0.9	63.9 ± 3.7	1.86 ± 0.13
CCPF	72.3 ± 2.4	85.6 ± 0.8	62.0 ± 3.8	1.90 ± 0.05
OND	75.0 ± 2.0	87.8 ± 0.9	69.1 ± 2.6	1.54 ± 0.07

TABLE VI
STATISTICAL ANALYSIS OF ROAD TRACKING RESULTS. ENTRIES SHOW THE p-VALUES OF THE WILCOXON MATCHED-PAIR SIGNED-RANK TEST

	input saving	distance saving	time saving	RMSE
CCKF-OND	0.055	0.016	0.023	0.016
CCPF-OND	0.055	0.008	0.016	0.008
CCKF-CCPF	0.742	0.461	0.195	0.461

differently for each Monte Carlo trial. For this reason, we evaluated the CCPF algorithm over ten Monte Carlo trials and report the average performance. The OND algorithm learned one predictor from each human input and performed novelty detection using the optimal candidate–predictor combination model.

A comparison of results obtained with the three algorithms CCKF, CCPF, and OND is given in Fig. 13. Table V shows average values and standard errors (defined as $\sigma_u/\sqrt{n}$, where σ_u is the standard deviation and $n = 8$ is the number of users). The results show an improvement of tracking performance using the proposed OND algorithm compared to the CCKF and CCPF algorithms. The OND algorithm is more efficient and more accurate than the other two, whereas CCKF and CCPF do not differ from each other. As shown in Table VI, all comparisons were tested using Wilcoxon matched-pair signed-rank test. The statistical results show that the differences between OND and CCKF and between OND and CCPF are significant, while the differences between CCKF and CCPF are not significant.

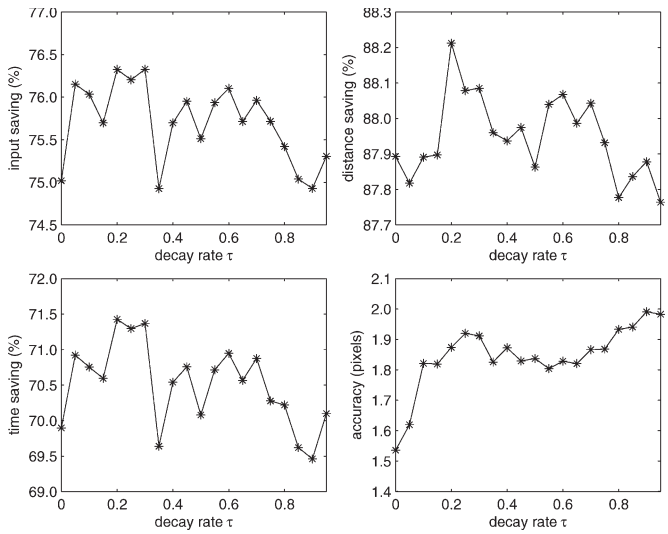


Fig. 14. Comparison of tracking performance with different decay rates in the learning session. The parameters for the model are given as follows: $\rho = 1$ and $\epsilon = 0.95$.

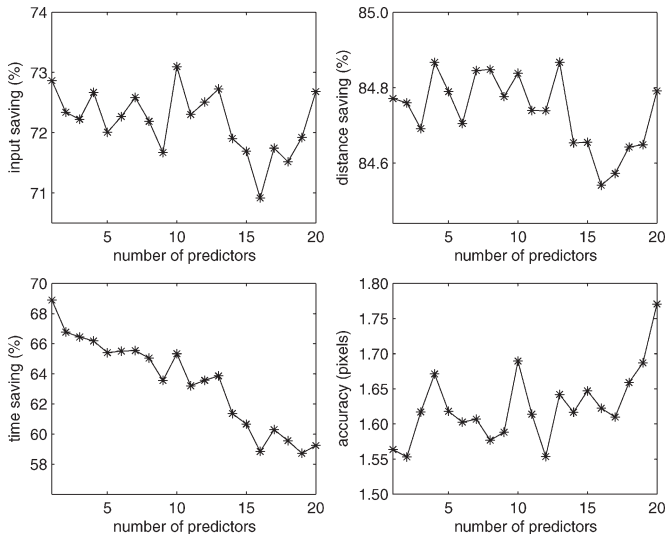


Fig. 15. Effects of acquiring multiple predictors from one human input. The parameters for the model are given as follows: $\tau = 0.2$, $\rho = 1$, and $\epsilon = 0.95$.

C. Discussion

To observe how the learning choices affect the overall performance of the semiautomatic road tracking, we performed experiments on learning models with different decay rates. Fig. 14 shows the results for one user; the results for the other users were similar. There was a trend for the system to achieve the highest tracking efficiency with a decay rate of about 0.2, but this occurred at the cost of a reduced tracking accuracy. In other words, an appropriate drifting model tends to be more efficient as it balances old and more recent training examples and, thus, is better able to characterize the gradual changes of the road feature. (A covariance analysis of the results revealed only marginally significant trends for input and distance savings, no effect for time saving, and a significant trend for accuracy.)

Fig. 15 shows the results of acquiring different numbers of predictors from one human input. Notice that the purpose

TABLE VII
COMPARISON OF TRACKING ALGORITHMS: TAI FOR THE OPTIMAL OBSERVATION–PREDICTOR COMBINATION MODEL, AND TAII FOR THE MULTIPLE-HYPOTHESES SUPPORT MODEL. PARAMETERS FOR THE MODELS ARE GIVEN AS FOLLOWS: $\tau = 0.05$, $\rho = 1$, AND $\epsilon = 0.99$. AVERAGE VALUES AND STANDARD ERRORS FOR EACH CRITERION ARE SHOWN

	input saving (%)	distance saving (%)	time saving (%)	RMSE (pixels)
TAI	75.2 ± 1.4	87.7 ± 1.0	68.3 ± 2.2	1.44 ± 0.07
TAII	76.2 ± 2.0	87.8 ± 1.0	71.2 ± 2.5	1.62 ± 0.07

TABLE VIII
COMPARISON OF TRACKING ALGORITHMS. ENTRIES SHOW THE p-VALUES OF THE WILCOXON MATCHED-PAIR SIGNED-RANK TEST

	input saving	distance saving	time saving	RMSE
TAI-TAII p-value	0.461	0.547	0.195	0.055

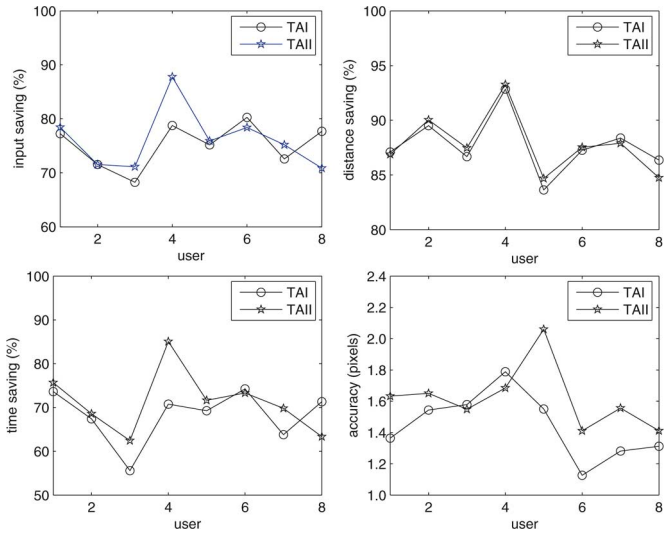


Fig. 16. Comparison of the tracking algorithms: TAI for the optimal observation–predictor combination model, and TAII for the multiple-hypotheses support model.

of prediction is to find the true road axis, hence tracking continues as long as a road axis can be found. As more predictors are learned from human inputs, the probability of getting at least one prediction on the road increases, allowing the automatic tracking to last longer. On the other hand, noise present on the road surface may affect the predictors, which in turn may generate false positive or false negative novelties in the tracking phase. An increase in the number of predictors also increases the probability that noise affects the model. This leads to a trade-off in the tracking results: as the number of predictors increases, the input efficiency and accuracy of the model decrease slowly, whereas time efficiency decreases fairly rapidly. (A covariance analysis of the results revealed significant trends for all four dependent measures, all $p < 0.05$.)

Finally, we compared the performance of two tracking algorithms, as shown in Tables VII and VIII, and Fig. 16. In both experiments, a single predictor was acquired from each human input during training. The results show that although the average efficiency of the multiple-hypotheses support model is slightly better than that of the optimal candidate–predictor combination model, the difference is statistically not significant. When noise is present on the road, the optimal candidate–predictor combination is likely to deviate from the true road axis point. It may cause more frequent tracking failures and requires more human involvement. With the weighted sum of multiple hypotheses, this deviation effect is slightly reduced.

VII. CONCLUSION

We have presented an online learning approach for novelty detection in image interpretation. This approach is applied to road tracking in aerial images within a human-guided framework that enables natural switching between human inputs and automatic tracking. It fills the gap between human and computer in image interpretation applications.

Depending on the requirements, single or multiple predictors can be learned from each human input. These predictors then automatically track a road using either an optimal candidate–predictor combination model or a multiple-hypotheses support model. We analyzed the theoretical and experimental difference between the learning and tracking models. Results show that multiple predictors from one human input further help in the learning. However, the system performance drops when too many predictors are acquired. Concerning tracking models, the performance of the multiple-hypotheses support model does not significantly differ from that of the optimal candidate–predictor combination model.

In addition to conceptual advantages, the experiments on real-world tasks validated the superior performance of the proposed approach, as compared to models without machine learning. Our approach is very generic and could be applied to similar applications that require intensive HCIs.

In our future work, we want to further explore our learning method. The proposed method uses novelty detection to learn road situations. It is a more natural approach to expand the learning to binary classes and to include off-road situation during the learning. Apart from road profiles, other road features can be used in the learning stage, such as road edges, image features in the frequency domain, and combinations of features. We also plan to apply the proposed method to satellite images, such IKONOS photographs [16], which have the same spatial resolution as the aerial photographs used in this work.

ACKNOWLEDGMENT

The authors would like to thank T. Caelli for helpful comments. NICTA is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Center of Excellence program.

REFERENCES

[1] M. Everingham, B. Thomas, and T. Troscianko, "Wearable mobility aid for low vision using scene classification in a Markov random field model framework," *Int. J. Hum.-Comput. Interact.*, vol. 15, no. 2, pp. 231–244, 2003.

[2] Z. Duric, W. Gray, R. Heishman, F. Li, A. Rosenfeld, M. Schoelles, C. Schunn, and H. Wechsler, "Integrating perceptual and cognitive modeling for adaptive and intelligent human–computer interaction," *Proc. IEEE*, vol. 90, no. 7, pp. 1272–1289, Jul. 2002.

[3] H. Koike, Y. Sato, and Y. Kobayashi, "Integrating paper and digital information on EnhancedDesk: A method for real time finger tracking on an augmented desk system," *ACM Trans. Comput.–Hum. Interact.*, vol. 8, no. 4, pp. 307–322, Dec. 2001.

[4] R. Cole, S. Vuuren, B. Pellom, K. Hacioglu, J. Ma, J. Movellan, S. Schwartz, D. Wade-Stein, W. Ward, and J. Yan, "Perceptive animated interfaces: First steps towards a new paradigm for human–computer interaction," *Proc. IEEE*, vol. 91, no. 9, pp. 1391–1405, Sep. 2003.

[5] J. Zhou, W. Bischof, and T. Caelli, "Human–computer interaction in map revision systems," in *Proc. 11th Int. Conf. Hum.-Comput. Interact.*, Las Vegas, NV, Jul. 2005. CD-ROM.

[6] M. Shin, L. Tsap, and D. Goldgof, "Gesture recognition using Bezier curves for visualization navigation from registered 3-D data," *Pattern Recognit.*, vol. 37, no. 5, pp. 1011–1024, 2004.

[7] C. Leubner, C. Brockmann, and H. Muller, "Computer-vision-based human–computer interaction with a back projection wall using arm gestures," in *Proc. 27th Euromicro Conf.*, Warsaw, Poland, 2001, pp. 308–314.

[8] A. Gee and R. Cipolla, "Fast visual tracking by temporal consensus," *Image Vis. Comput.*, vol. 14, no. 2, pp. 105–114, 1996.

[9] E. Ross, "Intelligent user interfaces: Survey and research directions," Dept. Comput. Sci., Univ. Bristol, Bristol, U.K., Tech. Rep. CSTR-00-004, 2000.

[10] D. McKeown and J. Denlinger, "Cooperative methods for road tracking in aerial imagery," in *Proc. IEEE Conf. Comput. Vis. and Pattern Recog.*, Ann Arbor, MI, Jun. 1988, pp. 662–672.

[11] G. Vosselman and J. Knecht, "Road tracing by profile matching and Kalman filtering," in *Proc. Workshop Autom. Extraction Man-Made Objects From Aerial and Space Images*, Apr. 1995, pp. 265–274.

[12] A. Gruen and H. Li, "Semi-automatic linear feature extraction by dynamic programming and LSB-snakes," *Photogramm. Eng. Remote Sens.*, vol. 63, no. 8, pp. 985–995, 1997.

[13] X. Hu, Z. Zhang, and C. Tao, "A robust method for semi-automatic extraction of road centerlines using a piecewise parabolic model and least square template matching," *Photogramm. Eng. Remote Sens.*, vol. 70, no. 12, pp. 1393–1398, 2004.

[14] T. Kim, S. Park, M. Kim, S. Jeong, and K. Kim, "Tracking road centerlines from high resolution remote sensing images by least squares correlation matching," *Photogramm. Eng. Remote Sens.*, vol. 70, no. 12, pp. 1417–1422, 2004.

[15] I. Couloigner and T. Ranchin, "Mapping of urban areas: A multiresolution modeling approach for semi-automatic extraction of streets," *Photogramm. Eng. Remote Sens.*, vol. 66, no. 7, pp. 867–874, 2000.

[16] R. Péteri and T. Ranchin, "Urban street mapping using Quickbird and IKONOS images," in *Proc. IEEE Int. Geosci. and Remote Sens. Symp.*, Toulouse, France, Jul. 2003, pp. 1721–1723.

[17] M. Amo, F. Martínez, and M. Torre, "Road extraction from aerial images using a region competition algorithm," *IEEE Trans. Image Process.*, vol. 15, no. 5, pp. 1192–1201, May 2006.

[18] A. Baumgartner, S. Hinz, and C. Wiedemann, "Efficient methods and interfaces for road tracking," *Int. Arch. Photogramm. Remote Sens.*, vol. 34, no. 3B, pp. 28–31, 2002.

[19] D. Xiong and J. Sperling, "Semiautomated matching for network database integration," *ISPRS J. Photogramm. Remote Sens.*, vol. 59, no. 1/2, pp. 35–46, 2004.

[20] J. Fails and D. Olsen, "Interactive machine learning," in *Proc. 8th Int. Conf. Intell. User Interface*, Miami, FL, 2003, pp. 39–45.

[21] M. Ware, E. Frank, G. Holmes, M. Hall, and I. Witten, "Interactive machine learning—Letting users build classifiers," *Int. J. Human–Comput. Stud.*, vol. 55, no. 3, pp. 281–292, 2001.

[22] P. Muneesawang and L. Guan, "An interactive approach for CBIR using a network of radial basis functions," *IEEE Trans. Multimedia*, vol. 6, no. 5, pp. 703–716, Oct. 2004.

[23] M. Maloof, P. Langley, T. Binford, R. Nevatia, and S. Sage, "Improved rooftop detection in aerial images with machine learning," *Mach. Learn.*, vol. 53, no. 1/2, pp. 157–191, Oct. 2003.

[24] A. Blum, "On-line algorithms in machine learning," in *Online Algorithms—The State of the Art*, A. Fiat and G. Woeginger, Eds. New York: Springer-Verlag, 1998, pp. 306–325.

[25] N. Littlestone and M. Warmuth, "The weighted majority algorithm," *Inf. Comput.*, vol. 108, no. 2, pp. 212–261, 1994.

[26] J. Kolter and M. Maloof, "Dynamic weighted majority: A new ensemble method for tracking concept drift," in *Proc. 3rd Int. Conf. Data Mining*, Los Alamitos, CA, Aug. 2003, pp. 123–130.

[27] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[28] M. Markou and S. Singh, "Novelty detection: A review—Part 2: Neural network based approaches," *Signal Process.*, vol. 83, no. 12, pp. 2499–2521, 2003.

[29] L. Cheng, S. Vishwanathan, D. Schuurmans, S. Wang, and T. Caelli, "Implicit online learning with kernels," in *Proc. NIPS*, Vancouver, BC, Canada, Dec. 2006.

[30] J. Zhou, W. Bischof, and T. Caelli, "Robust and efficient road tracking in aerial images," in *Int. Arch. Photogramm., Remote Sens. and Spatial Inf. Sci. (Proc. CMRT05)*, Vienna, Austria, Aug. 2005, vol. XXXVI, pp. 35–40. Part 3/W24.

[31] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.

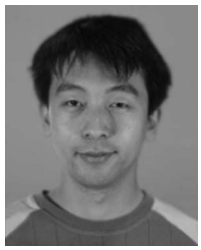
[32] J. Zhou, W. Bischof, and T. Caelli, "Understanding human–computer interactions in map revision," in *Proc. 10th Int. Workshop Struct. and Syntactic Pattern Recog.*, Lisbon, Portugal, Aug. 2004, pp. 287–295.

[33] C. Groat, "The national map—A continuing, critical need for the nation," *Photogramm. Eng. Remote Sens.*, vol. 69, no. 10, pp. 1087–1090, 2003.

[34] J. Zhou, W. Bischof, and T. Caelli, "Road tracking in aerial image based on human–computer interaction and Bayesian filtering," *ISPRS J. Photogramm. Remote Sens.*, vol. 61, no. 2, pp. 108–124, 2006.

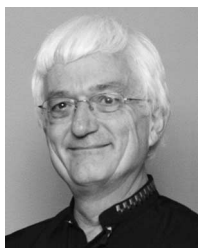
Jun Zhou received the B.S. degree in computer science and the B.E. degree in international business from the Nanjing University of Science and Technology, Nanjing, China, in 1996 and 1998, respectively, the M.S. degree in computer science from Concordia University, Montreal, QC, Canada, in 2002, and the Ph.D. degree from the University of Alberta, Edmonton, AB, Canada, in 2006.

He is currently a Researcher with the Canberra Laboratory, National ICT Australia, Canberra, Australia, and also with the Research School of Information Sciences and Engineering, Australian National University, Canberra. At the same time, he is an Adjunct Lecturer in the College of Engineering and Computer Science, Australian National University. His research interests include pattern recognition, remote sensing, human–computer interaction, and machine learning with human-in-the-loop.



Li Cheng received the B.S. degree from Jilin University, Changchun, China, the M.S. degree from Nankai University, Tianjin, China, and the Ph.D. degree from the University of Alberta, Edmonton, AB, Canada, in 2004.

He is currently a Researcher with the Canberra Laboratory, National ICT Australia, Canberra, Australia. He is also with the Research School of Information Sciences and Engineering, Australian National University, Canberra. His research interests are mainly on computer vision and machine learning.



Walter F. Bischof received the Ph.D. degree in psychology from the University of Bern, Bern, Switzerland, in 1982.

He is currently a Professor of computing science and the Co-Director of the Advanced Man–Machine Interface Laboratory, University of Alberta, Edmonton, AB, Canada. He is an Associate Editor of *Spatial Vision* and on the Editorial Board of several other journals. He has published more than 100 publications in refereed journals and conference proceedings as well as several books and book chapters. His research interests are in the areas of spatial navigation, human–machine interfaces, human visual perception, and attentional mechanisms.