Machine Vision and Applications manuscript No.
(will be inserted by the editor)

Automated Insect Identification through Concatenated Histograms of Local Appearance Features

Feature Vector Generation and Region Detection for Deformable Objects

Enrique Larios[1], Hongli Deng[3], Wei Zhang[3], Matt Sarpola[4], Jenny Yuen[7], Robert Paasch[4], Andrew Moldenke[5], David Lytle[6], Salvador Ruiz Correa[8], Eric Mortensen[3], Linda Shapiro[2], Tom Dietterich[3]

[1] University of Washington, Department of Electrical Engineering
[2] University of Washington, Department of Computer Science and Engineering
[3] Oregon State University, School of Electrical Engineering and Computer Science
[4] Oregon State University, Department of Mechanical Engineering
[5] Oregon State University, Department of Botany and Plant Pathology
[6] Oregon State University, Department of Zoology
[7] Massachusetts Institute of Technology, Computer Science and AI Laboratory
[8] Children's National Medical Center, Department of Diagnostic Imaging and Radiology

Received: date / Revised: date

Abstract This paper describes a computer vision approach to automated rapid-throughput taxonomic identification of stonefly larvae. The long-term goal of this research is to develop a cost-effective method for environmental monitoring based on automated identification of indicator species. Recognition of stonefly larvae is challenging because they are highly articulated, they exhibit a high degree of intraspecies variation in size and color, and some species are difficult to distinguish visually, despite prominent dorsal patterning. The stoneflies are imaged via an apparatus that manipulates the specimens into the field of view of a microscope so that images are obtained under highly repeatable conditions. The images are then classified through a process that involves (a) identification of regions of interest, (b) representation of those regions as SIFT vectors [1], (c) classification of the SIFT vectors into learned "features" to form a histogram of detected features, and (d) classification of the feature histogram via state-of-the-art ensemble classification algorithms. The steps (a) to (c) compose the concatenated feature histogram (CFH) method. We apply three region detectors for part (a) above, including a newly developed principal curvature-based region (PCBR) detector. This detector finds stable regions of high curvature via a watershed segmentation algorithm. We compute a separate dictionary of learned features for each region detector, and then concatenate the histograms prior to the final classification step.

We evaluate this classification methodology on a task of discriminating among four stonefly taxa, two of which, *Calineuria* and *Doroneuria*, are difficult even for experts to discriminate. The results show that the combination of all three detectors gives four-class accuracy of 82% and three-class accuracy (pooling *Calineuria* and *Doroneuria*) of 95%. Each region detector makes a valuable contribution. In particular, our new PCBR detector is able to discriminate *Calineuria* and *Doroneuria* much better than the other detectors.

Send offprint requests to: Tom Dietterich, 1145 Kelley Engineering Center, Corvallis, OR 97331-5501, USA, tgd@eecs.oregonstate.edu

Key words classification, object recognition, interest operators, region detectors, SIFT descriptor

1 Introduction

There are many environmental science applications that could benefit from inexpensive computer vision methods for automated population counting of insects and other small arthropods. At present, only a handful of projects can justify the expense of having expert entomologists manually classify field-collected specimens to obtain measurements of arthropod populations. The goal of our research is to develop general-purpose computer vision methods, and associated mechanical hardware, for rapid-throughput image capture, classification, and sorting of small arthropod specimens. If such methods can be made sufficiently accurate and inexpensive, they could

have a positive impact on environmental monitoring and ecological science [2–4].

The focus of our initial effort is the automated recognition of stonefly (Plecoptera) larvae for the biomonitoring of freshwater stream health. Stream quality measurement could be significantly advanced if an economically practical method were available for monitoring insect populations in stream substrates. Population counts of stonefly larvae and other aquatic insects inhabiting stream substrates are known to be a sensitive and robust indicator of stream health and water quality [5]. Because these animals live in the stream, they integrate water quality over time. Hence, they provide a more reliable measure of stream health than single-time-point chemical measurements. Aquatic insects are especially useful as biomonitors because (a) they are found in nearly all running-water habitats, (b) their large species diversity offers a wide range of responses to water quality change, (c) the taxonomy of most groups is well known and identification keys are available, (d) responses of many species to different types of pollution have been established, and (e) data analysis methods for aquatic insect communities are available [6]. Because of these advantages, biomonitoring using aquatic insects has been employed by federal, state, local, tribal, and private resource managers to track changes in river and stream health and to establish baseline criteria for water quality standards. Collection of aquatic insect samples for biomonitoring is inexpensive and requires relatively little technical training. However, the sorting and identification of insect specimens can be extremely time consuming and requires substantial technical expertise. Thus, aquatic insect identification is a major technical bottleneck for large-scale implementation of biomonitoring.

Larval stoneflies are especially important for biomonitoring because they are sensitive to reductions in water quality caused by thermal pollution, eutrophication, sedimentation, and chemical pollution. On a scale of organic pollution tolerance from 0 to 10, with 10 being the most tolerant, most stonefly taxa have a value of 0, 1, or 2 [5]. Because of their low tolerance to pollution, change in stonefly abundance or taxonomic composition is often the first indication of water quality degradation. Most biomonitoring programs identify stoneflies to the taxonomic resolution of Family, although when expertise is available Genus-level (and occasionally Species-level) identification is possible. Unfortunately, because of constraints on time, budgets, and availability of expertise, some biomonitoring programs fail to resolve stoneflies (as well as other taxa) below the level of Order. This results in a considerable loss of information and, potentially, in the failure to detect changes in water quality.

In addition to its practical importance, the automated recognition of stoneflies raises many fundamental computer vision challenges. Stonefly larvae are highly-articulated objects with many sub-parts (legs, antennae, tails, wing pads, etc.) and many degrees of freedom. Some taxa exhibit interesting patterns on their dorsal sides, but others are not patterned. Some taxa are distinctive, others are very difficult to identify. Finally, as the larvae repeatedly molt, their size and color change. Immediately after molting, they are light colored, and then they gradually darken. This variation in size, color, and pose means that simple computer vision methods that rely on placing all objects in a standard pose cannot be applied here. Instead, we need methods that can handle significant variation in pose, size, and coloration.

To address these challenges, we have adopted the bag-of-features approach [7–9]. This approach extracts a bag of region-based "features" from the image without regard to their relative spatial arrangement. These features are then summarized as a feature vector and classified via state-of-the-art machine learning methods. The primary advantage of this approach is that it is invariant to changes in pose and scale as long as the features can be reliably detected. Furthermore, with an appropriate choice of classifier, not all features need to be detected in order to achieve high classification accuracy. Hence, even if some features are occluded or fail to be detected, the method can still succeed. An additional advantage is that only weak supervision (at the level of entire images) is necessary during training.

A potential drawback of this approach is that it ignores some parts of the image, and hence loses some potentially useful information. In addition, it does not capture the spatial relationships among the detected regions. We believe that this loss of spatial information is unimportant in this application, because all stoneflies share the same body plan and, hence, the spatial layout of the detected features provides very little discriminative information.

The bag-of-features approach involves five phases: (a) region detection, (b) region description, (c) region classification into features, (d) combination of detected features into a feature vector, and (e) final classification of the feature vector. For region detection, we employ three different interest operators: (a) the Hessian-affine detector [10], (b) the Kadir entropy detector [11], and (c) a new detector that we have developed called the principal curvature-based region detector (PCBR). The combination of these three detectors gives better performance than any single detector or pair of detectors. The combination was critical to achieving good classification rates.

All detected regions are described using Lowe's SIFT representation [1]. At training time, a Gaussian mixture model (GMM) is fit to the set of SIFT vectors, and each mixture component is taken to define a feature. The GMM can be interpreted as a classifier that, given a new SIFT vector, can compute the mixture component most likely to have generated that vector. Hence, at classification time, each SIFT vector is assigned to the most likely feature (i.e., mixture component). A histogram consisting of the number of SIFT vectors assigned to each fea-

ture is formed. A separate GMM, set of features, and feature vector is created for each of the three region detectors and each of the stonefly taxa. These feature vectors are then concatenated prior to classification. The steps mentioned above form the concatenated feature histogram (CFH) method, which allows the use of general classifiers from the machine learning literature. The final labeling of the specimens is performed by an ensemble of logistic model trees [12], where each tree has one vote.

The rest of the paper is organized as follows. Section 2 discusses existing systems for insect recognition as well as relevant work in generic object recognition in computer vision. Section 3 introduces our PCBR detector and its underlying algorithms. In Section 4, we describe our insect recognition system including the apparatus for manipulating and photographing the specimens and the algorithms for feature extraction, learning, and classification. Section 5 presents a series of experiments to evaluate the effectiveness of our classification system and discusses the results of those experiments. Finally, Section 6 draws some conclusions about the overall performance of our system and the prospects for rapid-throughput insect population counting.

2 Related Work

We divide our discussion of related work into two parts. First, we review related work in insect identification methods. Then we discuss work in generic object recognition.

2.1 Automated Insect Identification Systems

A few other research groups have developed systems that apply computer vision methods to discriminate among a defined set of insect species.

2.1.1 Automated Bee Identification System (ABIS). The ABIS system [13] performs identification of bees based on features extracted from their forewings. Each bee is manually positioned and a photograph of its forewing is obtained in a standard pose. From this image, the wing venation is identified, and a set of key wing cells (areas between veins) are determined. These are used to align and scale the images. Then geometric features (lengths, angles, and areas) are computed. In addition, appearance features are computed from small image patches that have been smoothed and normalized. Classification is performed using Support Vector Machines and Kernel Discriminant Analysis.

This project has obtained very good results, even when discriminating between bee species that are known to be hard to classify. It has also overcome its initial requirement of expert interaction with the image for feature extraction; although it still has the restriction of complex user interaction to manipulate the specimen for the capture of the wing image. The ABIS feature extraction algorithm incorporates prior expert knowledge about wing venation. This facilitates the bee classification task; but makes it very specialized. This specialization precludes a straightforward application to other insect identification tasks.

2.1.2 Digital Automated Identification SYstem (DAISY). DAISY [14] is a general-purpose identification system that has been applied to several arthropod identification tasks including mosquitoes (*Culex p. molestus* vs. *Culex p. pipiens*), palaeartic ceratopogonid biting midges, ophionines (parasites of lepidoptera), parasitic wasps in the genus *Enicospilus*, and hawk-moths (Sphingidae) of the genus *Xylophanes*. Unlike our system, DAISY requires user interaction for image capture and segmentation, because specimens must be aligned in the images. This might hamper DAISY's throughput and make its application infeasible in some monitoring tasks where the identification of large samples is required.

In its first version, DAISY built on the progress made in human face detection and recognition via eigen-images [15]. Identification proceeded by determining how well a specimen correlated with an optimal linear combination of the principal components of each class. This approach was shown to be too computationally expensive and error-prone.

In its second version, the core classification engine is based on a random n-tuple classifier (NNC) [16] and plastic self organizing maps (PSOM). It employs a pattern to pattern correlation algorithm called the normalized vector difference (NVD) algorithm. DAISY is capable of handling hundreds of taxa and delivering the identifications in seconds. It also makes possible the addition of new species with only a small computational cost. On the other hand, the use of NNC imposes the requirement of adding enough instances of each species. Species with high intra-class variability require many training instances to cover their whole appearance range.

2.1.3 SPecies IDentification, Automated and web accessible (SPIDA-web). SPIDA-web [4] is an automated identification system that applies neural networks for species classification from wavelet encoded images. The prototype SPIDA-web system has been tested on the spider family Trochanteriidae (consisting of 119 species in 15 genera) using images of the external genitalia.

SPIDA-web's feature vector is built from a subset of the components of the wavelet transform using the Daubechines 4 function. The spider specimen has to be manipulated by hand, and the image capture, preprocessing and region selection also require direct user interaction. The images are oriented, normalized, and scaled into a 128x128 square prior to analysis. The specimens are classified in a hierarchical manner, first to genus and then to species. The classification engine is composed of

a trained neural network for each species in the group. Preliminary results for females indicate that SPIDA is able to classify images to genus level with 95-100% accuracy. The results of species-level classification still have room for improvement; most likely due to the lack of enough training samples.

2.1.4 Summary of previous insect identification work. This brief review shows that existing approaches rely on manual manipulation and image capture of each specimen. Some systems also require the user to manually identify key image features. To our knowledge, no system exists that identifies insects in a completely automated way, from the manipulation of the specimens to the final labeling. The goal of our research is to achieve full rapid-throughput automation, which we believe is essential to supporting routine bio-monitoring activities. One key to doing this is to exploit recent developments in generic object recognition, which we now discuss.

2.2 Generic Object Recognition

The past decade has seen the emergence of new approaches to object-class recognition based on region detectors, local features, and machine learning. These methods are able to recognize objects from images taken in non-controlled environments with variability in the position and orientation of the objects, with cluttered backgrounds, and with some degree of occlusion. Furthermore, these methods only require supervision at the level of whole images—the position and orientation of the object in each training image does not need to be specified. These approaches compare favorably with previous global-feature approaches, for example [17,18].

The local feature approaches begin by applying an interest operator to identify "interesting regions". These regions must be reliably detected in the sense that the same region can be found in images taken under different lighting conditions, viewing angles, and object poses. Further, for generic object recognition, these detected regions must be robust to variation from one object to another within the same generic class. Additionally, the regions must be informative—that is, they must capture properties that allow objects in different object classes to discriminate from one another. Special effort has been put into the development of affine-invariant region detectors to achieve robustness to moderate changes in viewing angle. Current affine-invariant region detectors can be divided into two categories: intensity-based detectors and structure-based detectors. The intensity-based region detectors include the Harris-corner detector [19], the Hessian-affine detector [20,10], the maximally stable extremal region detector (MSER) [21], the intensity extrema-based region detector (IBR) [22], and the entropy-based region detector [11]. Structure-based detectors include the edge-based region detector (EBR) [23] and the scale-invariant shape feature (SISF) detector [24].

Upon detection, each region must then be characterized as a vector of features. Several methods have been employed for this purpose, but by far the most widely-used region representation is David Lowe's 128-dimensional SIFT descriptor [1], which is based on histograms of local intensity gradients. Other region descriptors can be computed including image patches (possibly after smoothing and down-sampling), photometric invariants, and various intensity statistics (mean, variance, skewness, kurtosis).

Once the image has been converted into a collection of vectors—where each vector is associated with a particular region in the image—two general classes of methods have been developed for predicting the object class from this information. The first approach is known as the "bag of features" approach, because it disregards the spatial relationships among the SIFT vectors and treats them as an un-ordered bag of feature vectors. The second approach is known as the "constellation method", because it attempts to capture and exploit the spatial relationships among the detected regions. (Strictly speaking, the term constellation model refers to the series of models developed by Burl, Weber and Perona [25].)

In the bag-of-features approach, the standard method is to take all of the SIFT vectors from the training data and cluster them (possibly preceded by a dimensionality-reduction step such as PCA). Each resulting cluster is taken to define a "keyword", and these keywords are collected into a codebook or dictionary [26–28]. The dictionary can then be applied to map each SIFT vector into a keyword, and therefore, to map the bag of SIFT features into a bag of keywords.

The final step of our approach is to train a classifier to assign the correct class label to the bag of keywords. The most direct way to do this is to convert the bag into a feature vector and apply standard machine learning methods such as AdaBoost [29]. One simple method is to compute a histogram where the i-th element corresponds to the number of occurrences in the image of the i-th keyword.

Another classification strategy is to employ distance-based learning algorithms such as the nearest-neighbor method. This involves defining a distance measure between two bags of keywords such as the minimum distance between all keywords from one bag and all keywords from the other bag.

Given a new image to classify, the process of finding interesting regions, representing them as SIFT vectors, mapping those to keywords, and classifying the resulting bags of keywords is repeated.

In the constellation method, several techniques have been applied for exploiting the spatial layout of the detected regions. The star-shaped model [30,31] is a common choice, because it is easy to train and evaluate. Fergus et al. [32] employ a generative model of the (x, y)

distribution of the regions as a 2-dimensional Gaussian distribution. More complex methods apply discriminative graphical models to capture the relations between the detected regions [33–35].

3 Principal Curvature-Based Region Detector

Before describing our stonefly recognition system, we first introduce our new Principal Curvature-Based Region (PCBR) detector. This detector is of independent interest and we have demonstrated elsewhere that it can be applied to a wide range of object recognition problems [36].

The PCBR detector grew out of earlier experiments that apply Steger's "curvilinear" detector [37] to the stonefly images. The curvilinear detector finds line structures (either curved or straight) such as roads in aerial or satellite images or blood vessels in medical scans. When applied to stonefly images, the detector provides a kind of sketch of the characteristic patterning that appears on the insects' dorsal side. Further, these curvilinear structures can be detected over a range of viewpoints, scales, and illumination changes.

However, in order to produce features that readily map to image regions, which can then be used to build a descriptor (such as SIFT), our PCBR detector ultimately uses only the first steps of Steger's curvilinear detector process—that of computing the principal eigenvalue of the Hessian matrix at each pixel. We note that since both the Hessian matrix and the related second moment matrix quantify a pixel's local image geometry, they have also been applied in several other interest operators such as the Harris [19], Harris-affine [38], and Hessian-affine [10] detectors to find image positions where the local image geometry is changing in more than one direction. Likewise, Lowe's maximal difference-of-Gaussian (DoG) detector [1] also uses components of the Hessian matrix (or at least approximates the sum of the diagonal elements) to find points of interest. However, we also note that our PCBR detector is quite different from these other methods. Rather than finding interest "points", our method applies a watershed segmentation to the principal curvature image to find "regions" that are robust to various image transformations. As such, our PCBR detector combines differential geometry—as used by the Harris- and Hessian-affine interest point detectors—with concepts found in region-based structure detectors such as EBR [23] or SISF [24].

3.1 A Curvature-Based Region Detector

Given an input image (Figure 1a), our PCBR region detector can be summarized as follows:

1. Compute the Hessian matrix image describing each pixel's local image curvature.
2. Form the principal curvature image by extracting the largest positive eigenvalue from each pixel's Hessian matrix (Figure 1b).
3. Apply a gray scale morphological closing on the principal curvature image to remove noise and threshold the resulting image to obtain a "clean" binary principal curvature image (Figure 1c).
4. Segment the clean image into regions using the watershed transform (Figures 1d and 1e).
5. Fit an ellipse to each watershed regions to produce the detected interest regions (Figure 1f).

Each of these steps is detailed in the following paragraphs.

3.2 Principal Curvature Image

There are two types of structures that have high curvature in one direction: edges and curvilinear structures. Viewing an image as an intensity surface, the curvilinear structure detector looks for ridges and valleys of this surface. These correspond to white lines on black backgrounds or black lines on white backgrounds. The width of the detected line is determined by the Gaussian scale used to smooth the image (see Eq. 1 below). Ridges and valleys have large curvature in one direction, edges have high curvature in one direction and low curvature in the orthogonal direction, and corners (or highly curved ridges and valleys) have high curvature in two directions. The shape characteristics of the surface can be described by the Hessian matrix, which is given by

$$\mathbf{H}(\mathbf{x}, \sigma_D) = \begin{bmatrix} I_{xx}(\mathbf{x}, \sigma_D) & I_{xy}(\mathbf{x}, \sigma_D) \\ I_{xy}(\mathbf{x}, \sigma_D) & I_{yy}(\mathbf{x}, \sigma_D) \end{bmatrix} \quad (1)$$

where I_{xx}, I_{xy} and I_{yy} are the second-order partial derivatives of the image and σ_D is the Gaussian scale at which the second partial derivatives of the image are computed. The interest point detectors mentioned previously [19, 38, 10] apply the Harris measure (or a similar metric [1]) to determine a point's saliency. The Harris measure is given by

$$det(\mathbf{A}) - k \cdot tr^2(\mathbf{A}) > threshold \quad (2)$$

where *det* is the determinant, *tr* is the trace, and the matrix $\mathbf{A}$ is either the Hessian matrix, $\mathbf{H}$, (for the Hessian-affine detector) or the second moment matrix,

$$\mathbf{M} = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}, \quad (3)$$

for the Harris or Harris-affine detectors. The constant k is typically between 0.03 and 0.06 with 0.04 being very common. The Harris measure penalizes (i.e., produces

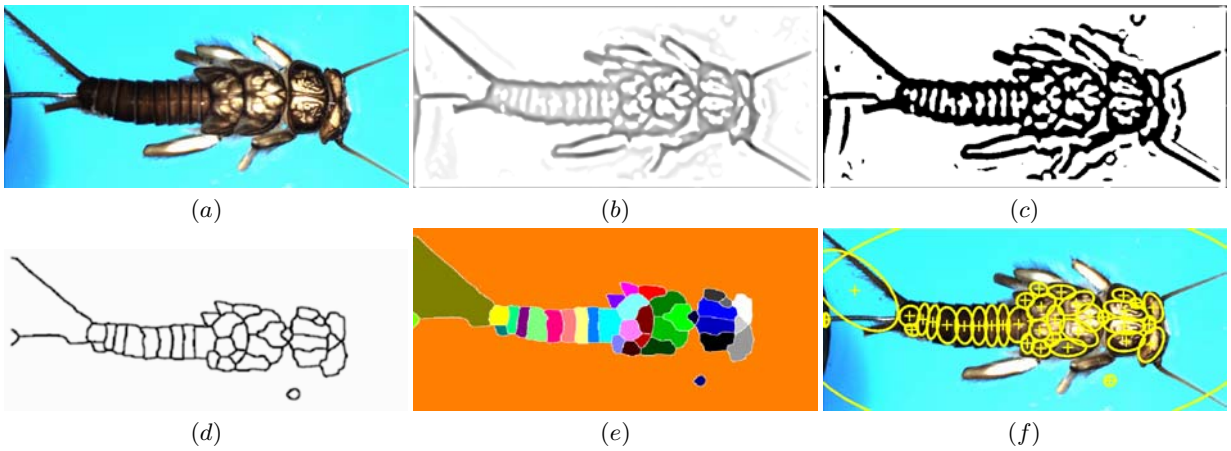


Fig. 1 Regions defined by principal curvature. (a) The original, (b) principal curvature, and (c) cleaned binary images. The resulting (d) boundaries and (e) regions that result by applying the watershed transform to (c). (f) The final detected regions created by fitting an ellipse to each region.

low values for) "long" structures for which the first or second derivative in one particular orientation is very small. One advantage of the Harris metric is that it does not require explicit computation of the eigenvalue or eigenvectors. However, computing the eigenvalues and eigenvectors for a 2×2 matrix requires only a single Jacobi rotation to eliminate the off-diagonal term, I_{xy}, as noted by Steger [37].

Our PCBR detector complements the previous interest point detectors. We abandon the Harris measure and exploit those very long structures as detection cues. The principal curvature image is given by either

$$P(\mathbf{x}) = \max(\lambda_1(\mathbf{x}), 0) \tag{4}$$

or

$$P(\mathbf{x}) = \min(\lambda_2(\mathbf{x}), 0) \tag{5}$$

where $\lambda_1(\mathbf{x})$ and $\lambda_2(\mathbf{x})$ are the maximum and minimum eigenvalues, respectively, of H at $\mathbf{x}$. Eq. 4 provides a high response only for dark lines on a light background (or on the dark side of edges) while Eq. 5 is used to detect light lines against a darker background. We do not take the largest absolute eigenvalue since that would produce two responses for each edge. For our stonefly project, we have found that the patterning on the stonefly dorsal side is better characterized by the dark lines and as such we apply Eq. 4. Figure 1(b) shows an eigenvalue image that results from applying Eq. 4 to the grayscale image derived from Fig. 1(a). We utilize the principle curvature image to find the stable regions via watershed segmentation [39].

3.3 Watershed Segmentation

Our detector depends on a robust watershed segmentation. A main problem with segmentation via the watershed transform is its sensitivity to noise and image variations. Figure 2(a) shows the result of applying the watershed algorithm directly to the eigenvalue image (shown in Fig. 1(b)). Many of the small regions are due to noise or other small, unstable image variations. To achieve a more stable watershed segmentation, we first apply a grayscale morphological closing followed by hysteresis thresholding. The grayscale morphological closing operation is defined as

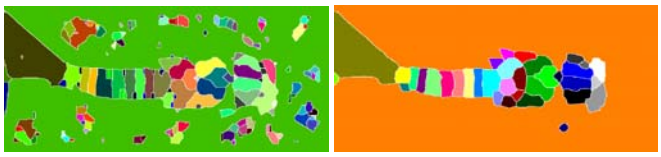


Fig. 2 (a) Watershed segmentation of original eigenvalue image (Fig. 1b). (b) Detection results using the "clean" principal curvature image (Fig. 1c).

$$f \bullet b = (f \oplus b) \ominus b \tag{6}$$

where f is the image (P from Eq. 4 for our application), b is a disk-shaped structuring element, and $\oplus$ and $\ominus$ are the grayscale dilation and erosion, respectively. The closing operation removes the small "potholes" in the principal curvature terrain, thus eliminating many local minima that result from noise and would otherwise produce watershed catchment basins.

However, beyond the small (in terms of area of influence) local minima, there are other minima that have larger zones of influence and are not reclaimed by the morphological closing. Some of these minima should indeed be minima since they have a very low principal curvature response. However, other minima have a high response but are surrounded by even higher peaks in the principle curvature terrain. A primary cause for these high "dips" between ridges is that the Gaussian scale

used to compute the Hessian matrix is not large enough to match the thickness of the line structure; hence the second derivative operator produces principal curvature responses that tend toward the center of the thick line but don't quite meet up. One solution to this problem is to use a multiscale approach and try to estimate the best scale to apply at each pixel. Unfortunately, this would require that the Hessian be applied at many scales to find the single characteristic scale for each pixel. Instead, we choose to compute the Hessian at just a few scales ($\sigma_D = 1, 2, 4$) and then use eigenvector-flow hysteresis thresholding to fill in the gaps between scales.

In eigenvalue-flow hysteresis thresholding, there is both a high and a low threshold—just as in traditional hysteresis thresholding. For this application, we have set the high threshold at 0.04 to indicate strong principal curvature response. Pixels with a strong response act as seeds that expand out to include connected pixels that are above the low threshold. Unlike traditional hysteresis thresholding, our low threshold is a function of the support each pixel's major eigenvector receives from neighboring pixels. Of course, we want the low pixel to be high enough to avoid over-segmentation and low enough to prevent ridge lines from fracturing. As such, we choose our low threshold on a per-pixel basis by comparing the direction of the major (or minor) eigenvector to the direction of the adjacent pixels' major (or minor) eigenvectors. This can be done by simply taking the absolute value of the inner (or dot) product of a pixel's normalized eigenvector with that of each neighbor. The inner product is 1 for vectors pointing in the same direction and 0 for orthogonal vectors. If the average dot product over all neighbors is high enough, we set the low to high threshold ratio to 0.2 (giving an absolute threshold of $0.04 \cdot 0.2 = 0.008$); otherwise the low to high ratio is 0.7 (for an absolute low threshold of 0.028). These ratios were chosen based on experiments with hundreds of stonefly images.

Figure 3 illustrates how the eigenvector flow supports an otherwise weak region. The red arrows are the major eigenvectors and the yellow arrows are the minor eigenvectors. To improve visibility, we draw them at every 4 pixels. At the point indicated by the large white arrow, we see that the eigenvalue magnitudes are small and the ridge there is almost invisible. Nonetheless, the direction of the eigenvectors are quite uniform. This eigenvector-based active thresholding process yields better performance in building continuous ridges and in filling in scale gaps between ridges, which results in more stable regions (Fig. 2(b)).

The final step is to perform the watershed transform on the clean binary image. Since the image is binary, all black (or 0-valued) pixels become catchment basins and the midline of the thresholded white ridge pixels potentially become watershed lines if it separates two distinct catchment basins. After performing the watershed transform, the resulting segmented regions are fit with ellipses, via PCA, that have the same second-moment as these watershed regions. These ellipses then define the final interest regions of the PCBR detector (Fig. 1(f)).

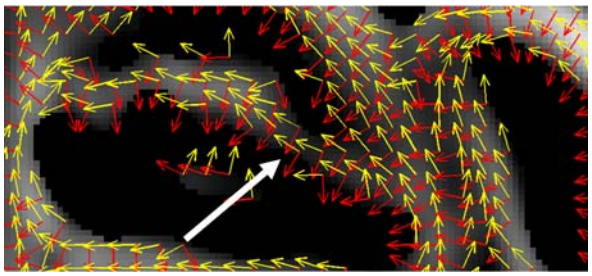


Fig. 3 Illustration of how the eigenvector flow is used to support weak principal curvature response.

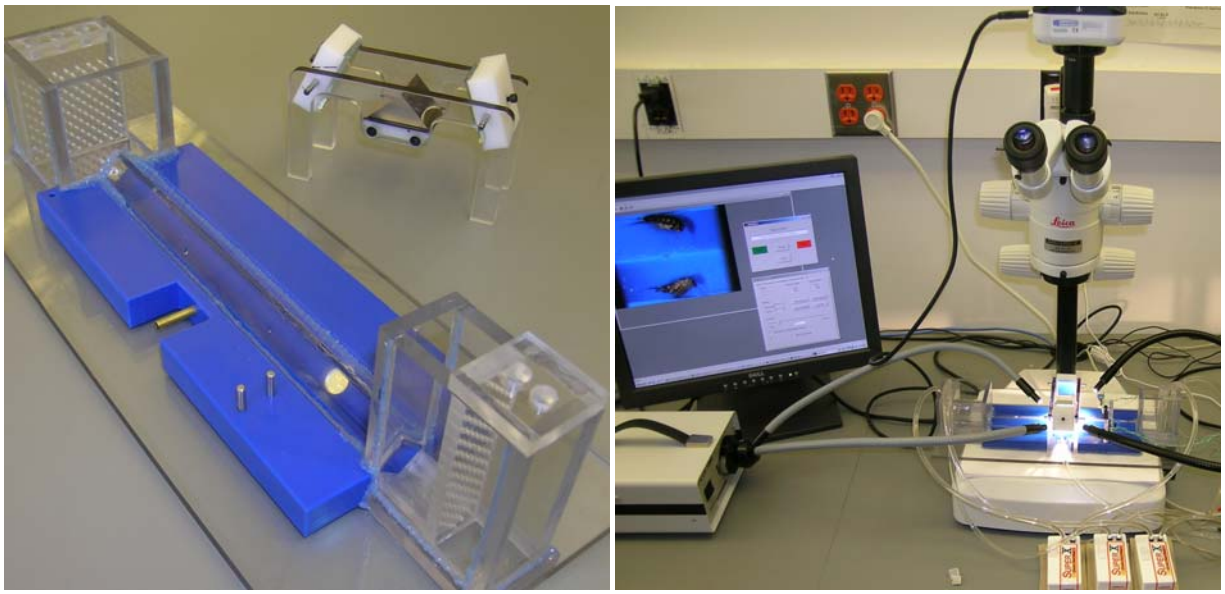
4 Stonefly Identification System

The goal of our work is to provide a rapid-throughput system for classifying stonefly larvae to the species level. To achieve this, we have developed a system that combines a mechanical apparatus for manipulating and photographing the specimens with a software system for processing and classifying the resulting images. We now describe each of these components in turn.

4.1 Semi-Automated Mechanical Manipulation and Imaging of Stonefly Larvae

The purpose of the hardware system is to speed up the image capture process in order to make bio-monitoring viable and to reduce variability during image capture. To achieve consistent, repeatable image capture, we have designed and constructed a software-controlled mechanical stonefly larval transport and imaging apparatus that positions specimens under a microscope, rotates them to obtain a dorsal view, and photographs them with a high-resolution digital camera.

Figure 4 shows the mechanical apparatus. The stoneflies are kept in alcohol (70% ethanol) at all times, and therefore, the apparatus consists of two alcohol reservoirs connected by an alcohol-filled tube (having a diamond cross-section). To photograph a specimen, it is manually inserted into the arcylic well shown at the right edge of the figure and then pumped through the tube. Infrared detectors positioned part way along the tube detect the passage of the specimen and cut off the pumps. Then a side fluid jet "captures" the specimen in the field of view of the microscope. When power to this jet is cut off, the specimen settles to the bottom of the tube where it can be photographed. The side jet can be activated repeatedly to spin the specimen to obtain different views.



(a) (b)

Fig. 4 (a) Prototype mirror and transportation apparatus. (b) Entire stonefly transportation and imaging setup (with microscope and attached digital camera, light boxes, and computer controlled pumps for transporting and rotating the specimen.

Once a suitable image has been obtained (a decision currently made by the human operator), the specimen is then pumped out of the tube and into the plexiglass well at the left edge of the figure. For this project, a "suitable image" is one that gives a good back (dorsal side) view of the specimen. In future work, we plan to construct a "dorsal view detector" to automatically determine when a good dorsal image has been obtained. In addition, future versions of the apparatus will physically sort each specimen into an appropriate bin based on the output of the recognizer.

Figure 4(b) shows the apparatus in place under the microscope. Each photograph taken by the camera captures two images at a 90 degree separation via a set of mirrors. The original purpose of this was to support 3D reconstruction of the specimens, but for the work described in this paper, it doubles the probability of obtaining a good dorsal view in each shot.

All images are captured using a QImaging MicroPublisher 5.0 RTV 5 megapixel color digital camera. The digital camera is attached to a Leica MZ9.5 high-performance stereo microscope at 0.63x magnification. We use a 0.32 objective on the microscope to increase the field of view, depth of field, and working distance. Illumination is provided by gooseneck light guides powered by Volpi V-Lux 1000 cold light sources. Diffusers installed on the guides reduce glare, specular reflections, and hard shadows. Care was taken in the design of the apparatus to minimize the creation of bubbles in the alcohol, as these could confuse the recognizer.

With this apparatus, we can image a few tens of specimens per hour. Figure 6 shows some example images obtained using this stonefly imaging assembly.

Table 1 Dictionary Construction. D is the number of region detectors (3 in our case), and K is the number of stonefly taxa to be recognized (4 in our case)

Dictionary Construction

For each detector $d = 1, \ldots, D$
  For each class $k = 1, \ldots, K$
    Let $S_{d,k}$ be the set of SIFT vectors that results from applying detector d to all cluster images from class k.
    Fit a Gaussian mixture model to $S_{d,k}$ to obtain a set of mixture components $\{C_{d,k,\ell}\}, \ell = 1, \ldots, L$.
    The GMM estimates the probability of each SIFT vector $\mathbf{s} \in S_{d,k}$ as

$$P(\mathbf{s}) = \sum_{\ell=1}^{L} C_{d,k,\ell}(\mathbf{s} \mid \mu_{d,k,\ell}, \mathbf{\Sigma}_{d,k,\ell}) P(\ell).$$

    where $C_{d,k,\ell}$ is a multi-variate Gaussian distribution with mean $\mu_{d,k,\ell}$ and diagonal covariance matrix $\mathbf{\Sigma}_{d,k,\ell}$.

    Define the keyword mapping function $key_{d,k}(\mathbf{s}) = \text{argmax}_\ell \; C_{d,k,\ell}(\mathbf{s} \mid \mu_{d,k,\ell}, \mathbf{\Sigma}_{d,k,\ell})$

4.2 Training and Classification

Our approach to classification of stonefly larvae follows closely the "bag of features" approach but with several modifications and extensions. Figure 5 gives an overall picture of the data flow during training and classification, and Tables 1, 2, and 3 provide pseudo-code for our method. We now provide a detailed description.

The training process requires two sets of images, one for defining the dictionaries and one for training the clas-

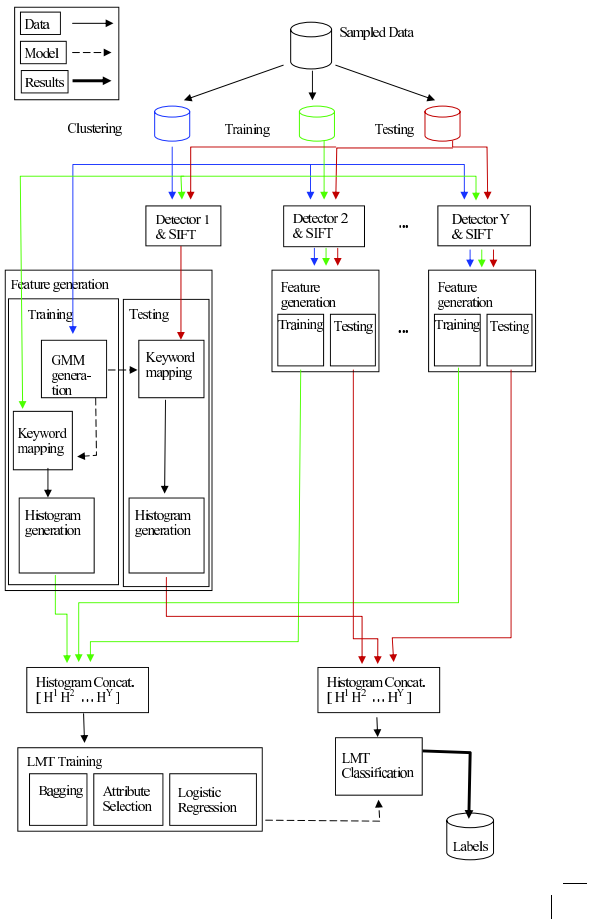


Fig. 5 Object recognition system overview: Feature generation and classification components

Table 2 Feature Vector Construction

Feature Vector Construction

To construct a feature vector for an image:
For each detector $d = 1, \ldots, D$
  For each class $k = 1, \ldots, K$
    Let $H_{d,k}$ be the keyword histogram for detector d and class k
    Initialize $H_{d,k}[\ell] = 0$ for $\ell = 1, \ldots, L$
    For each SIFT vector $\mathbf{s}$ detected by detector d
      increment $H_{d,k}[key_{d,k}(\mathbf{s})]$
Let H be the concatenation of the $H_{d,k}$ histograms for all d and k.

Table 3 Training and Classification. B is the number of bootstrap iterations (i.e., the size of the classifier ensemble).

Training

Let $T = \{(H_i, y_i)\}, i = 1, \ldots, N$ be the set of N training examples where H_i is the concatenated histogram for training image i and y_i is the corresponding class label (i.e., stonefly species).
For bootstrap replicate $b = 1, \ldots, B$
  Construct training set T_b by sampling N training examples randomly with replacement from T
  Let LMT_b be the logistic model tree fitted to T_b

Classification

Given a test image, let H be the concatenated histogram resulting from feature vector construction.
Let $votes[k] = 0$ be the number of votes for class k.
For $b = 1, \ldots, B$
  Let $\hat{y}_b$ be the class predicted by LMT_b applied to H.
  Increment $votes[\hat{y}_b]$.
Let $\hat{y} = \text{argmax}_k\ votes[k]$ be the class with the most votes.
Predict $\hat{y}$.

sifier. In addition, to assess the accuracy of the learned classifier, we need a holdout test data set, as usual. Therefore, we begin by partitioning the data at random into three subsets: clustering, training, and testing.

As mentioned previously, we apply three region detectors to each image: (a) the Hessian-affine detector [10], (b) the Kadir Entropy detector [11], and (c) our PCBR detector. We use the Hessian-affine detector implementation available from Mikolajczyk[1] with a detection threshold of 1000. For the Kadir entrophy detector, we use the binary code made available by the author[2] and set the scale search range between $25 - 45$ pixels with the saliency threshold at 58. All the parameters for the two detectors mentioned above are obtained empirically by modifying the default values in order to obtain reasonable regions. For the PCBR detector, we detect in three scales with $\sigma_D = 1, 2, 4$. The higher value in hysteresis thresholding is 0.04. The two ratios applied to get the lower thresholds are 0.2 and 0.7—producing low thresholds of 0.008 and 0.028, respectively. Each detected region is represented by a SIFT vector using Mikolajczyk's modification to the binary code distributed by David Lowe [1].

We then construct a separate dictionary for each region detector d and each class k. Let $S_{d,k}$ be the SIFT descriptors for the regions found by detector d in all cluster-set images from class k. We fit a Gaussian mixture model (GMM) to $S_{d,k}$ via the Expectation-Maximization (EM) algorithm. A GMM with L components has the form

$$p(\mathbf{s}) = \sum_{\ell=1}^{L} C_{d,k,\ell}(\mathbf{s} \mid \mu_{d,k,\ell}, \mathbf{\Sigma}_{d,k,\ell}) P(\ell) \qquad (7)$$

where $\mathbf{s}$ denotes a SIFT vector and the component probability distribution $C_{d,k,\ell}$ is a multivariate Gaussian density function with mean $\mu_{d,k,\ell}$ and covariance matrix $\mathbf{\Sigma}_{d,k,\ell}$ (constrained to be diagonal). Each fitted component of the GMM defines one of L keywords. Given a

[1] www.robots.ox.ac.uk/~vgg/research/affine/
[2] www.robots.ox.ac.uk/~timork/salscale.html

new SIFT vector $\mathbf{s}$, we compute the corresponding keyword $\ell = key_{d,k}(s)$ by finding the ℓ that maximizes $p(\mathbf{s} \mid \mu_{d,k,\ell}, \mathbf{\Sigma}_{d,k,\ell})$. Note that we disregard the mixture probabilities $P(\ell)$. This is equivalent to mapping $\mathbf{s}$ to the nearest cluster center μ_ℓ under the Mahalobis distance defined by Σ_ℓ.

We initialize EM by fitting each GMM component to each cluster obtained by the k-means algorithm. The k-means algorithm is initialized by picking random elements. The EM algorithm iterates until the change in the fitted GMM error from the previous iteration is less than 0.05% or until a defined number of iterations is reached. In practice, learning of the mixture almost always reaches the first stopping criterion (the change in error is less that 0.05%).

After building the keyword dictionaries, we next construct a set of training examples by applying the three region detectors to each training image. We characterize each region found by detector d with a SIFT descriptor and then map the SIFT vector to the nearest keyword (as describe above) for each class k using $key_{d,s}$. We accumulate the keywords to form a histogram $H_{d,k}$ and concatenate these histograms to produce the final feature vector. With D detectors, K classes, and L mixture components, the number of attributes A in the final feature vector (i.e., the concatenated histogram) is $D \cdot K \cdot L$.

Upon constructing the set of training examples, we next learn the classifier. We employ a state-of-the-art ensemble classification method: bagged logistic model trees. Bagging [40] is a general method for constructing an ensemble of classifiers. Given a set T of labeled training examples and a desired ensemble size B, it constructs B bootstrap replicate training sets T_b, $b = 1, \ldots, B$. Each bootstrap replicate is a training set of size $|T|$ constructed by sampling uniformly with replacement from T. The learning algorithm is then applied to each of these replicate training sets T_b to produce a classifier LMT_b. To predict the class of a new image, each LMT_b is applied to the new image and the predictions vote to determine the overall classification. The ensemble of LMTs classifier only interacts with the feature vectors generated by the CFH method.

Our chosen learning algorithm is the logistic model tree (LMT) method of Landwehr, Hall, and Frank [12]. An LMT has the structure of a decision tree where each leaf node contains a logistic regression classifier. Each internal node tests the value of one chosen feature from the feature vector against a threshold and branches to the left child if the value is less than the threshold and to the right child if the value is greater than or equal to the threshold. LMTs are fit by the standard top-down divide-and-conquer method employed by CART [41] and C4.5 [42]. At each node in the decision tree, the algorithm must decide whether to introduce a split at that point or make the node into a leaf (and fit a logistic regression model). This choice is made by a one-step lookahead search in which all possible features and thresholds are evaluated to see which one will result in the best improvement in the fit to the training data. In standard decision trees, efficient purity measures such as the GINI index or the information gain can be employed to predict the quality of the split. In LMTs, it is instead necessary to fit a logistic regression model to the training examples that belong to each branch of the proposed split. This is computationally expensive, although the expense is substantially reduced via a clever incremental algorithm based on logit-boost [43]. Thorough benchmarking experiments show that LMTs give robust state-of-the-art performance [12].

Table 4 Specimens and images employed in the study

Taxon	Specimens	Images
Calineuria	85	400
Doroneuria	91	463
Hesperoperla	58	253
Yoraperla	29	124

5 Experiments and Results

We now describe the series of experiments carried out to evaluate our system. We first discuss the data set and and show some example images to demonstrate the difficulty of the task. Then we present the series of experiments and discuss the results.

5.1 Stonefly Dataset

We collected 263 specimens of four stonefly taxa from freshwater streams in the mid-Willamette Valley and Cascade Range of Oregon: the species *Calineuria californica* (Banks), the species *Doroneuria baumanni* Stark & Baumann, the species *Hesperoperla pacifica* (Banks), and the genus *Yoraperla*. Each specimen was independently classified by two experts, and only specimens that were classified identically by both experts were considered in the study. Each specimen was placed in its own vial with an assigned control number and then photographed using the apparatus described in Section 4. Approximately ten photos were obtained of each specimen, which yields 20 individual images. These were then manually examined, and all images that gave a dorsal view within 30 degrees of vertical were selected for analysis. Table 4 summarizes the number of specimens and dorsal images obtained.

A potential flaw in our procedure is that the specimen vials tended to be grouped together by taxon (i.e., several *Calineurias* together, then several *Doroneurias*, etc.), so that in any given photo session, most of the specimens being photographed belong to a single taxon. This could introduce some implicit cues (e.g., lighting, bubbles, scratches) that might permit the learning algorithm to "cheat". The apparatus constrains the lighting

so that it is very consistent in all sessions. We did detect some bubbles in the images. In cases where the region detectors found those bubbles, we manually remove the detections to ensure that they are not influencing the results.

Figure 6 shows some of the images collected for the study. Note the variety of colors, sizes, and poses. Note also that *Yoraperla*, which is in the family Peltoperlidae, is quite distinctive in color and shape. The other three taxa, which are all in the family Perlidae, are quite similar to each other, and the first two (*Calineuria* and *Doroneuria*) are exceedingly difficult to distinguish. This is emphasized in Figure 7, which shows closeup dorsal views. To verify the difficulty of discriminating these two taxa, we conducted an informal study that tested the ability of humans to separate *Calineuria* and *Doroneuria*. A total of 26 students and faculty from Oregon State University were allowed to train on 50 randomly-selected images of *Calineuria* and *Doroneuria*, and were subsequently tested with another 50 images. Most of the subjects (21) had some prior entomological experience. The mean score was 78.6% correctly identified (std. dev. = 8.4). There was no statistical difference between the performance of entomologists and non-entomologists (Wilcoxon two-sample test [44], $W = 57.5$, $p \leq 0.5365$).

Given the characteristics of the taxa, we defined three discrimination tasks, which we term CDHY, JtHY, and CD as follows:

CDHY: Discriminate among all four taxa.

JtHY: Merge *Calineuria* and *Doroneuria* to define a single class, and then discriminate among the resulting three classes.

CD: Focus on discriminating only between *Calineuria* and *Doroneuria*.

The CDHY task assesses the overall performance of the system. The JtHY task is most relevant to biomonitoring, since *Calineuria* and *Doroneuria* have identical pollution tolerance levels. Hence, discriminating between them is not critical for our application. Finally, the CD task presents a very challenging objective recognition problem, so it is interesting to see how well our method can do when it focuses only on this two-class problem.

Performance on all three tasks is evaluated via three-fold cross-validation. The images are randomly partitioned into three equal-sized sets under the constraint that all images of any given specimen were required to be placed in the same partition. In addition, to the extent possible, the partitions are stratified so that the class frequencies are the same across the three partitions. Table 5 gives the number of specimens and images in each partition.

In each "fold" of the cross-validation, one partition serves as the clustering data set for defining the dictionaries, a second partition serves as the training data set, and the third partition serves as the test set.

Table 5 Partitions for 3-fold cross-validation.

Partition	# Specimens	# Images
1	87	413
2	97	411
3	79	416

Our approach requires specification of the following parameters:

- the number L of mixture components in the Gaussian mixture model for each dictionary,
- the number B of bootstrap replicates for bagging,
- the minimum number M of training examples in the leaves of the logistic model trees, and
- the number I of iterations of logit boost employed for training the logistic model trees.

These parameters are set as follows. L is determined through a series of EM fitting procedures for each species. We increment the number of mixture components until the GMM is capable of modeling the data distribution—when the GMM achieves a relative fitting error below 5% in less than 100 EM iterations. The resulting values of L are 90, 90, 85 and 65 for *Calineuria*, *Doroneuria*, *Hesperoperla*, and *Yoraperla*, respectively. Likewise, B is determined by evaluating a series of bagging ensembles with different numbers of classifiers on the same training set. The number of classifiers in each ensemble is incremented by two until the training error starts to increase, at which point B is simply assigned to be five less than that number. The reason we assign B to be 5 less than the number that causes the training error to increase—rather than simply assign it to the largest number that produces the lowest error—is that the smaller number of boostrap replicates helps to avoid overfitting. Table 6 shows the value of B for each of the three tasks. The minimum number M of instances that each leaf in the LMT requires to avoid pruning is set to 15, which is the default value for the LMT implementation recommended by the authors. The number of logit boost iterations I is set by internal cross-validation within the training set while the LMT is being induced.

Table 6 Number of bagging iterations for each experiments.

Experiment	Bagging Iterations B
4-species: CDHY	20
3-species: JtHY	20
2-species: CD	18

(a) (b) (c) (d)

Fig. 6 Example images of different stonefly larvae species. (a) *Calineuria*, (b) *Doroneuria*, (c) *Hesperoperla* and (d) *Yoraperla*.

(a) (b) (c) (d)

Fig. 7 Images visually comparing *Calineuria* and *Doroneuria*. (a) *Calineuria*, (b) *Doroneuria*, (c) *Calineuria* detail and (d) *Doroneuria* detail.

5.2 Results

We designed our experiments to achieve two goals. First, we wanted to see how well the CFH method (with three region detectors) coupled with an ensemble of LMTs performs on the three recognition tasks. To establish a basis for evaluation, we also apply the method of Opelt, et al. [45], which is currently one of the best object recognition systems. Second, we wanted to evaluate how each of the three region detectors affects the performance of the system. To achieve this second goal, we train our system using 7 different configurations corresponding to training with all three detectors, all pairs of detectors, and all individual detectors.

5.2.1 Overall Results Table 7 shows the classification rates achieved by the CFH method on the three discrimination tasks. Tables 8, 9, and 10 show the confusion matrices for the three tasks. On the CDHY task, our system achieves 82% correct classifications. The confusion matrix shows that it achieves near perfect recognition of *Yoraperla*. It also recognizes *Hesperoperla* very well with only a few images misclassified as *Calineuria* or *Doroneuria*. As expected, the main difficulty is to discriminate *Calineuria* and *Doroneuria*. When these two classes are pooled in the JtHY task, performance reaches 95% correct, which is excellent. It is interesting to note that if we had applied the four-way classifier and then pooled the *predictions* of the classifiers, the 3-class performance would have been slightly better (95.48% versus 95.08%). The difference is that in the JtHY task, we learn a combined dictionary for the merged *Calineuria* and *Doroneuria* (CD) class, whereas in the 4-class task, each taxon has its own dictionaries.

A similar phenomenon occurs in the 2-class CD task. Our method attains 79% correct classification when trained on only these two tasks. If instead, we applied the CDHY classifiers and treated predictions for *Hesperoperla* and *Yoraperla* as errors, the performance would be

Table 7 Percentage of images correctly classified by our system with all three region detectors along using a 95% confidence interval.

Task	Accuracy [%]
CDHY	82.42 ± 2.12
JtHY	95.40 ± 1.16
CD	79.37 ± 2.70

Table 8 CDHY confusion matrix of the combined Kadir, Hessian-affine and PCBR detectors

predicted as ⇒	*Cal.*	*Dor.*	*Hes.*	*Yor.*
Calineuria	315	79	6	0
Doroneuria	80	381	2	0
Hesperoperla	24	22	203	4
Yoraperla	1	0	0	123

Table 9 JtHY confusion matrix of the combined Kadir, Hessian-affine and PCBR detectors

predicted as ⇒	*Joint CD*	*Hes.*	*Yor.*
Joint CD	857	5	1
Hesperoperla	46	203	4
Yoraperla	0	1	123

Table 10 CD confusion matrix of the combined Kadir, Hessian-affine and PCBR detectors

predicted as ⇒	*Calineuria*	*Doroneuria*
Calineuria	304	96
Doroneuria	82	381

slightly better (79.61% versus 79.37%). These differences are not statistically significant, but they do suggest that in future work it might be useful to build separate dictionaries and classifiers for groups within each taxon (e.g., first cluster by size and color) and then map the resulting predictions back to the 4-class task. On this binary classification task, our method attains 79% correct classification, which is approximately equal to the mean for human subjects with some prior experience.

Our system is capable of giving a confidence measure to each of the existing categories. We performed a series of experiments where the species assignment is thresholded by the difference between the two highest confidence measures. In this series, we vary the threshold from 0 to 1. If the difference is higher than the defined threshold, the label of the highest is assigned otherwise the specimen is declared as "uncertain". Figure 8 shows the plotting of the accuracy against the rejection rate. The curves show us that if we reject around 30% of the specimens, all the tasks will reach an accuracy higher than 90%, even the CD task.

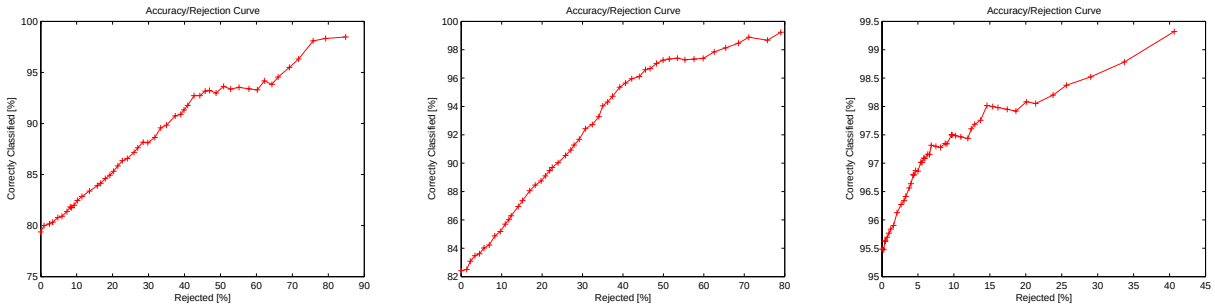
We also evaluate the performance of our classification methodology relative to a competing method [45] on the most difficult CD task using the same image features. Opelt's method is similar to our method in that it is also based on ensemble learning principles (AdaBoost), and it is also capable of combining multiple feature types for classification. We adapted Opelt's Matlab implementation to our features and used the default parameter settings given in the paper. The Euclidean distance metric was used for the SIFT features and number of iterations I was set to 100. Table 11 summarizes the classification rates. Our system provides 8 to 12% better accuracy than Opelt's method for all four combinations of detectors. In addition, training Opelt's classifier is more computationally expensive than is training our system. In particular, the complexity of computing Opelt's feature-to-image distance matrix is $O(T^2R^2D)$, where T is the number of training images, R is the maximum number of detected image regions in a single image, and $D = 128$ is the SIFT vector dimension. The total number of detected training regions, $T \cdot R$, is easily greater than $20,000$) in this application. On the other hand, training our system is much faster. The complexity of building the LMT ensemble classifier (which dominates the training computation) is $O(T \cdot A \cdot I)$, where A is the number of histogram attributes and I is the number of LMT induction iterations (typically in the hundreds).

Table 11 Comparison of CD classification rates using Opelt's method and our system with different combinations of detectors. A $\surd$ indicates the detector(s) used.

Hessian Affine	Kadir Entropy	PCBR	Accuracy[%]	
			Opelt [45]	CFH & LMTs
$\surd$			60.59	70.10
	$\surd$		62.63	70.34
		$\surd$	67.86	79.03
$\surd$	$\surd$	$\surd$	70.10	79.37

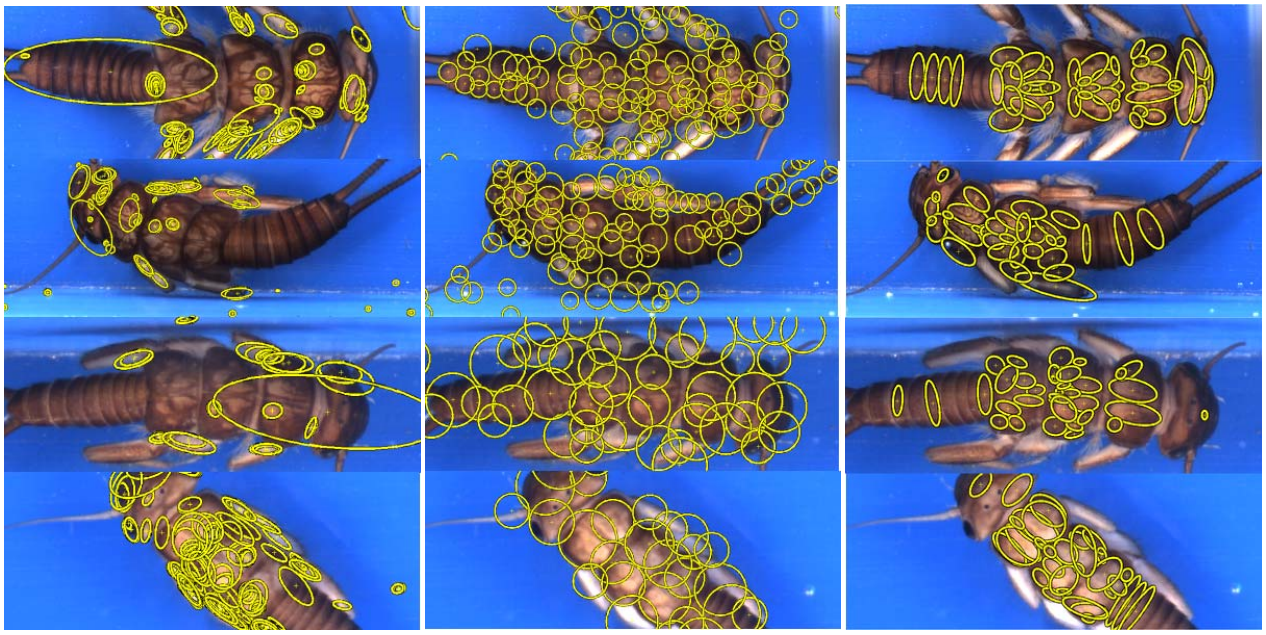
5.2.2 Results for Multiple Region Detectors Table 12 summarizes the results of applying all combinations of one, two, and three detectors to the CDHY, JtHY, and CD tasks. The first three lines show that each detector has unique strengths when applied alone. The Hessian-affine detector works best on the 4-class CDHY task; the Kadir detector is best on the 3-class JtHY task, and the PCBR detector gives the best 2-class CD results. On the pairwise experiments it appears that the Hessian-affine and PCBR complement each other well. The best pairwise results for the JtHY task is obtained by the Kadir-Hessian pair; which appears to be better for tasks that require an overall assessment of shape. Finally, the combination of all three detectors gives the best results on each task.

To understand the region detector results, it is helpful to look at their behaviors. Figures 9 and 10 show the regions found by each detector on selected *Calineuria* and *Doroneuria* specimens. The detectors behave in quite different ways. The PCBR detector is very stable, although it does not always identify all of the relevant regions. The Kadir detector is also stable, but it finds a very large number of regions, most of which are not rele-



CD *CDHY* *JtHY*

Fig. 8 Accuracy/Rejection curves for the three experiments with all the detectors combined while changing the confidence-difference threshold



(*a*) (*b*) (*c*)

Fig. 9 Visual Comparison of the regions output by the three detectors on three *Calineuria* specimens. (a) Hessian-affine, (b) Kadir Entropy, (c) PCBR

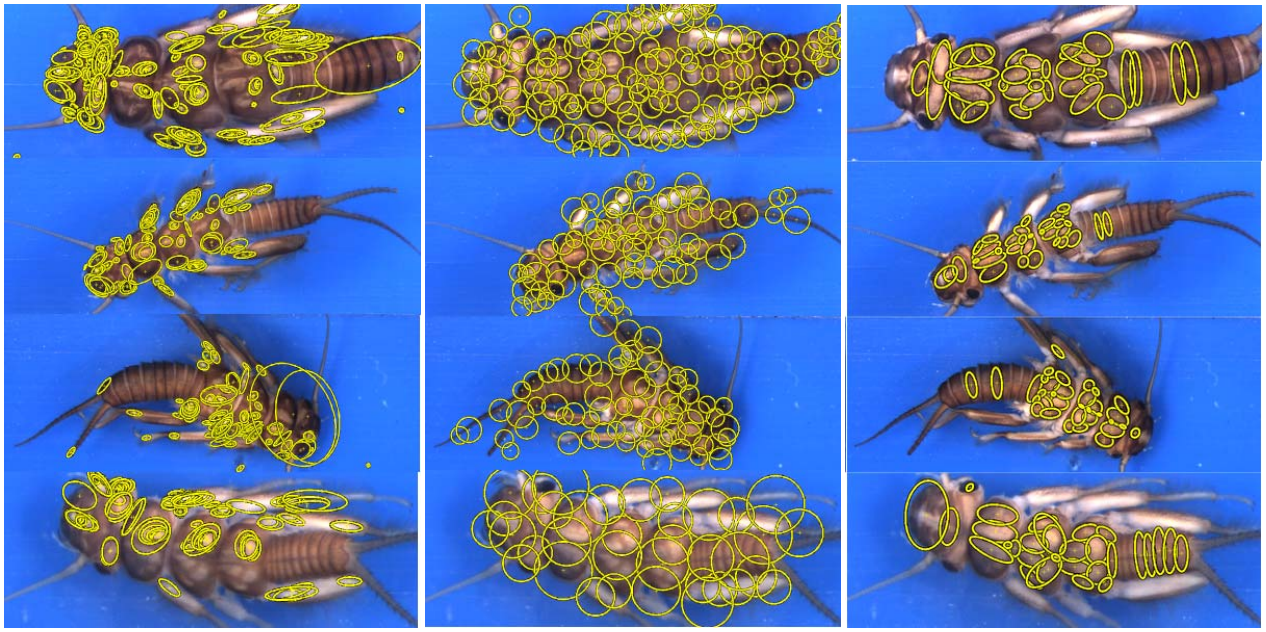
Table 12 Classification rates using our system with different combinations of detectors. A $\surd$ indicates the detector(s) used.

Hessian Affine	Kadir Entropy	PCBR	Accuracy[%]		
			CDHY	JtHY	CD
$\surd$			73.14	90.32	70.10
	$\surd$		70.64	90.56	70.34
		$\surd$	71.69	86.21	79.03
$\surd$	$\surd$		78.14	94.19	74.16
$\surd$		$\surd$	80.48	93.79	78.68
	$\surd$	$\surd$	78.31	92.09	68.83
$\surd$	$\surd$	$\surd$	82.42	95.40	79.37

vant. The Hessian-affine detector finds very good small-scale regions, but its larger-scale detections are not useful for classification. The PCBR detector focuses on the interior of the specimens, whereas the other detectors (especially Kadir) tend to find points on the edges between the specimens and the background. In addition to concentrating on the interior, the regions found by the PCBR detector are more "meaningful" in that they correspond better to body parts. This may explain why the PCBR detector did a better job on the CD task.

6 Conclusions and Future Work

This paper has presented a combined hardware-software system for rapid-throughput classification of stonefly larvae. The goal of the system is to perform cost-effective bio-monitoring of freshwater streams. To this end, the mechanical apparatus is capable of nearly unassisted manipulation and imaging of stonefly specimens while also obtaining consistently high quality images. The generic object recognition algorithms attain classification accuracy that is sufficiently good (82% for 4-classes; 95% for



(a) (b) (c)

Fig. 10 Visual Comparison of the regions output by the three detectors on four *Doroneuria* specimens. (a) Hessian-affine, (b) Kadir Entropy, (c) PCBR

3-classes) to support the application. By rejecting for manual classification the specimens in which the confidence level is not high enough; only a reasonable 30% of the samples would require further processing while the remaining identified specimens can reach an accuracy above 90% on all the defined tasks.

We compared our CFH method to Opelt's related state-of-art method on the most difficult task, discriminating *Calineuria* from *Doroneuria*. The CFH method always achieved better performance. It is also worth noticing that human subjects with some prior experience and using the same images reached an accuracy equal to our method. Finally, we described a new region detector, the principal curvature-based region (PCBR) detector. Our experiments demonstrated that PCBR is particularly useful for discriminating between the two visually similar species, and, as such provides an important contribution in attaining greater accuracy.

There are a few details that must be addressed before the system is ready for field testing. First, the mechanical apparatus must be modified to include mechanisms for sorting the specimens into bins after they have been photographed and classified. Second, we need to develop an algorithm for determining whether a good dorsal image of the specimen has been obtained. We are currently exploring several methods for this including training the classifier described in this paper for the task. Third, we need to evaluate the performance of the system on a broader range of taxa. A practical bio-monitoring system for the Willamette Valley will need to be able to recognize around 8 stonefly taxa. Finally, we need to develop methods for dealing with specimens that are not stoneflies or that do not belong to any of the taxa that the system is trained to recognize. We are studying SIFT-based density estimation techniques for this purpose.

Beyond freshwater stream bio-monitoring, there are many other potential applications for rapid-throughput arthropod recognition systems. One area that we are studying involves automated population counts of soil mesofauna for soil biodiversity studies. Soil mesofauna are small arthropods (mites, spiders, pseudo-scorpions, etc.) that live in soils. There are upwards of 2000 species, and the study of their interactions and population dynamics is critical for understanding soil ecology and soil responses to different land uses. In our future work, we will test the hypothesis that the methods described in this paper, when combined with additional techniques for shape analysis and classification, will be sufficient to build a useful system for classifying soil mesofauna.

Acknowledgments

We wish to thank Andreas Opelt for providing the Matlab code of his PAMI'06 method for the comparison experiment. We also wish to thank Asako Yamamuro and Justin Miles for their assistance with the dataset stonefly identification.

References

1. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision **60** (2004) 91–110

2. Gaston, K.J., O'Neill, M.A.: Automated species identification: why not? Philosophical Transactions of the Royal Society B: Biological Sciences **359** (2004) 655–667
3. Hopkins, G.W., Freckleton, R.P.: Declines in the numbers of amateur and professional taxonomists: implications for conservation. Animal Conservation **5** (2002) 245–249
4. Do, M., Harp, J., Norris, K.: A test of a pattern recognition system for identification of spiders. Bulletin of Entomological Research **89** (1999) 217–224
5. Hilsenhoff, W.L.: Rapid field assessment of organic pollution with a family level biotic index. Journal of the North American Benthological Society **7** (1988) 65–68
6. Carter, J., Resh, V., Hannaford, M., Myers, M.: Macroinvertebrates as biotic indicators of env. qual. In: F. Hauer and G. Lamberti, editors. Methods in Stream Ecology, San Diego, Academic Press (2006)
7. G.Csurka, Bray, C., Fan, C.L.: Visual categorization with bags of keypoints. ECCV workshop (2004)
8. Dorko, G., Schmid, C.: Object class recognition using discriminative local features. PAMI submitted (2004)
9. Opelt, A., Fussenegger, M., Pinz, A., Auer, P.: Weak hypotheses and boosting for generic object detection and recognition. In: 8th European Conference on Computer Vision. Volume 2., Prague, Czech Republic (2004) 71–84
10. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. IJCV **60** (2004) 63–86
11. Kadir, T., Zisserman, A., Brady, M.: An affine invariant salient region detector. In: European Conference on Computer Vision (ECCV04). (2004) 228–241
12. Landwehr, N., Hall, M., Frank, E.: Logistic model trees. Mach. Learn. **59** (2005) 161–205
13. Arbuckle, T., Schroder, S., Steinhage, V., Wittmann, D.: Biodiversity informatics in action: identification and monitoring of bee species using ABIS. In: Proc. 15th Int. Symp. Informatics for Environmental Protection. Volume 1., Zurich (2001) 425–430
14. O'Neill, M.A., Gauld, I.D., Gaston, K.J., Weeks, P.: Daisy: an automated invertebrate identification system using holistic vision techniques. In: Proc. Inaugural Meeting BioNET-INTERNATIONAL Group for Computer-Aided Taxonomy (BIGCAT), Egham (2000) 13–22
15. Turk, M.A., Pentland, A.P.: Face recognition using eigenfaces. In: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition. (1991) 586–591
16. Lucas, S.: Face recognition with continuous n-tuple classifier. In: Proc. British Machine Vision Conference, Essex (1997) 222–231
17. Papageorgiou, C., Poggio, T.: A trainable system for object detection. Int. J. Comput. Vision **38** (2000) 15–33
18. Sung, K.K., Poggio, T.: Example-based learning for view-based human face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 39–51
19. Harris, C., Stephens, M.: A combined corner and edge detector. Alvey Vision Conference (1988) 147–151
20. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. ECCV **1** (2002) 128–142
21. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing **22** (2004) 761–767
22. Tuytelaars, T., Gool, L.V.: Wide baseline stereo matching based on local, affinely invariant regions. BMVC (2000) 412–425
23. Tuytelaars, T., Gool, L.V.: Matching widely separated views based on affine invariant regions. IJCV **59** (2004) 61–85
24. Jurie, F., Schmid, C.: Scale-invariant shape features for recognition of object categories. CVPR **2** (2004) 90–96
25. Burl, M., M.Weber, Perona, P.: A probabilistic approach to object recognition using local photometry and global geometry. In: Proc. ECCV. (1998) 628–641
26. Csurka, G., Dance, C., Fan, L., Williamowski, J., Bray, C.: Visual categorization with bags of keypoints. ECCV'04 workshop on Statistical Learning in Computer Vision (2004) 59–74
27. Dorkó, G., Schmid, C.: Object class recognition using discriminative local features. Accepted under major revisions to IEEE Transactions on Pattern Analysis and Machine Intelligence, updated 13 September (2005)
28. Jurie, F., Triggs, B.: Creating efficient codebooks for visual recognition. In: ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Washington, DC, USA, IEEE Computer Society (2005) 604–610
29. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Conference on Machine Learning. (1996) 148–156
30. Leibe, B., Seemann, E., Schiele, B.: Pedestrian detection in crowded scenes. In: CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, Washington, DC, USA, IEEE Computer Society (2005) 878–885
31. Shotton, J., Blake, A., Cipolla, R.: Contour-based learning for object detection. In: ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Washington, DC, USA, IEEE Computer Society (2005) 503–510
32. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Volume 2., Madison, Wisconsin (2003) 264–271
33. Bouchard, G., Triggs, B.: Hierarchical part-based visual object categorization. In: IEEE Conference on Computer Vision & Pattern Recognition. (2005) I 710–715
34. Kumar, S., August, J., Hebert, M.: Exploiting inference for approximate parameter learning in discriminative fields: An empirical study. In: 5th International Workshop, EMMCVPR 2005, St. Augustine, Florida, Springer-Verlag (2005) 153 – 168
35. Quattoni, A., Collins, M., Darrell, T.: Conditional random fields for object recognition. In: Proc. NIPS 2004, Cambridge, MA, MIT Press (2005)
36. Zhang, W., Deng, H., Dietterich, T.G., Mortensen, E.N.: A hierarchical object recognition system based on multi-scale principal curvature regions. International Conference of Pattern Recognition (2006) 1475–1490
37. Steger, C.: An unbiased detector of curvilinear structures. PAMI **20** (1998) 113–125
38. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A comparison of affine region detectors. IJCV (2005)

39. Vincent, L., Soille, P.: Watersheds in digital spaces: An efficient algorithm based on immersion simulations. PAMI **13** (1991) 583–598
40. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140
41. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Chapman & Hall, New York, NY, USA (1984)
42. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1993)
43. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting (1998)
44. Sokal, R.R., Rohlf, F.J.: Biometry. 3rd edn. W. H. Freeman & Co. (1995)
45. Opelt, A., Pinz, A., Fussenegger, M., Auer, P.: Generic object recognition with boosting. IEEE Trans. Pattern Anal. Mach. Intell. **28** (2006) 416–431