Title
Detecting Object Boundaries Using Low-, Mid-, and High-level Information

Authors
Songfeng Zheng
Zhuowen Tu
Alan L. Yuille

Publication Date
2011-10-25

Detecting Object Boundaries using Low-,Middle-, and High-Level Information.

SongFeng Zheng
Department of Statistics
University of California at Los Angeles
Los Angeles, CA 90095

Zhuowen Tu
Laboratory of NeuroImaging
University of California at Los Angeles
Los Angeles, CA 90095

Alan Yuille
Department of Statistics
University of California at Los Angeles
Los Angeles, CA 90095
yuille@stat.ucla.edu

In proccedings CVPR 2007. (CDROM - paperless proceedings).18-23 June. 2007.

Detecting Object Boundaries Using Low-, Mid-, and High-level Information

Songfeng Zheng
Dept. of Statistics, UCLA
sfzheng@stat.ucla.edu

Zhuowen Tu
Lab of Neuro Imaging, UCLA
zhuowen.tu@loni.ucla.edu

Alan L. Yuille
Dept. of Statistics, UCLA
yuille@stat.ucla.edu

Abstract

Object boundary detection and segmentation is a central problem in computer vision. The importance of combining low-level, mid-level, and high-level cues has been realized in recent literature. However, it is unclear how to efficiently and effectively engage and fuse different levels of information. In this paper, we emphasize a learning based approach to explore different levels of information, both implicitly and explicitly. First, we learn low-level cues for object boundaries and interior regions using a probabilistic boosting tree (PBT) [17, 6]. Second, we learn short and long range context information based on the results from the first stage. Both stages implicitly contain object-specific information such as texture and local geometry, and it is shown that this implicit knowledge is extremely powerful. Third, we use high-level shape information explicitly to further refine the object segmentation and to parse the object into components. The algorithm is trained and tested on a challenging dataset of horses [2], and the results obtained are very encouraging compared with other approaches. In detailed experiments we show significantly better performance (e.g. F-values of 0.75 compared to 0.66) than the best comparable reported performance on this dataset [14]. Furthermore, the system only needs 1.5 minutes for a typical image. Although our system is illustrated on horse images, the approach can be directly applied to detecting/segmenting other types of objects.

1. Introduction

Object boundary detection and segmentation is a key problem in computer vision. It has been well accepted that low-level cues such as classical edge detectors are insufficient to perform this task. For example, Fig. 1 shows the results of the Canny edge detector [3] when applied to horse images [2]. It is a nontrivial problem to detect the horse boundary from the edge map. Marr conceived an outline of a solution for this problem [13] by combining low-, mid-, and high-level cues. Despite promising work in this direction [4, 21, 5, 15], no complete solution has been established so far.

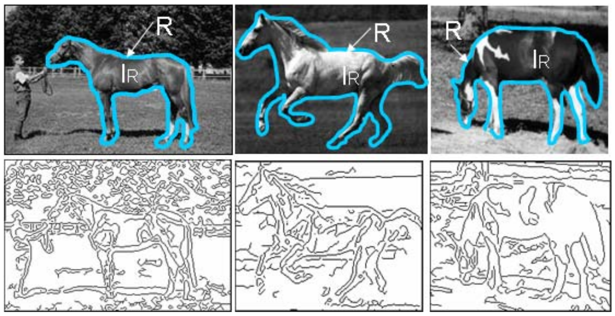


Figure 1. Illustration of three training images. The first row shows three typical images, and each contains a horse in the center, where R is the boundary we want to detect and I_R denotes the foreground region. The second row displays edges detected by Canny edge detector at scale $\sigma = 1.0$.

Recently, the problem of combining low-,mid-, and high-level information for this task has become more tractable due to progress in machine learning and statistics. Borenstein et al. [2] combined top-down information (configuration on learned image patches) and bottom-up approaches (segmentation based on intensity) for figure-ground segmentation. In the image parsing framework [18], data-driven techniques (bottom-up) were used to guide generative (top-down) inference. Fergus et al. [8] built a top-down model based on features extracted by interest point operators. Conditional Markov random fields models [10, 16] have been used to enforce local consistency between neighboring structures. Combining both top-down and bottom-up learning in a loop is emphasized in [20]. OBJCUT [12] also combined different levels of information and performs segmentation by graph cuts.

These approaches have shown the promise of combining low-level and high-level information. However, when, where and how to combine different channels of information is yet unclear. For example, the image parsing algorithm [18] assumes that generative models are available for modelling the appearance of objects, but no existing generative models are able to capture the complex appearance patterns of the horses in Fig. 1. The patches used in [2] cannot deal with large scale changes and they also have difficulties in capturing the complex variations in appearance. Some approaches like [8, 10, 16] rely on representing features and are unable to locate object boundaries. Other ap-

1-4244-1180-7/07/$25.00 ©2007 IEEE

proaches, while effective, lead to complex models which require solving time consuming inference problems [20, 12].

In this paper, we use learning based methods to combine different levels of information both implicitly and explicitly. This results in simple models which require only simple inference and can be computed in about one minute. We rely on the probabilistic boosting tree (PBT) [17, 6] algorithm to learn models for low-level and mid-level cues while implicitly taking high-level context into account, then we use shape matching [19] to supply explicit high-level information and to parse the horses into head, back, legs, and other parts. Stacking [23] builds classifiers on top of other classifiers and, thus, is slightly related to our approach. However, stacking is a general term rather than a specific algorithm.

We compare our system with other approaches which have tackled this problem. The most directly comparable approach is the work of Ren et al. [14] which gives detailed performance evaluations for combining low-,mid-, and high- level information. Our results show significant improvements at all levels. It is less easy to make direct comparison with other works [2, 12, 20] because some of them were not evaluated on large testing datasets, and the details of performance evaluation were not given for others. Also some approaches [12, 20] used color images, which are less challenging than gray-scale images we are using. Our approach is a very simple and clear one, and it only needs about 1.5 minutes on an ordinary PC for a typical image, while speed is not reported in other works.

2. Problem Formulation

In this section, we give the formulation of our approach. Given a gray-scale image, we assume there is an object of interest in the foreground and our task is to automatically detect the boundary of that object, and thus, perform foreground and background segmentation. In addition, we want to parse the object and identify its parts (e.g. head, leg, back, etc.).

2.1. The ideal Bayesian formulation

For an input image $\mathbf{I}$, the task of foreground/background segmentation is to infer which pixels belong to the background and which belong to the foreground. A solution W can be denoted as

$$W = \{(R_k, \theta_k), k \in \{0, 1\}\}$$

where R_1 is the region for the foreground and θ_1 denotes its corresponding model parameter; R_0 denotes the background region. A region consists of a set of connected pixels. Equivalently, a region can be denoted by its boundaries since one can always be derived from the other. We have $R_0 \bigcup R_1 = \Lambda$ and $R_0 \cap R_1 = \emptyset$, where Λ defines the 2D lattice of the input $\mathbf{I}$, which is the set of all the pixels. The optimal solution W^* can be inferred in the Bayesian framework as

$$\begin{aligned} W^* &= \arg\max_W \, p(W|\mathbf{I}) \\ &= \arg\max_W \, p(\mathbf{I}(R_0)|R_0, \theta_0) p(\mathbf{I}(R_1)|R_1, \theta_1) p(R_1) \end{aligned}$$

where $p(\mathbf{I}(R_0)|R_0, \theta_0)$ and $p(\mathbf{I}(R_1)|R_1, \theta_1)$ define the appearance models for the background and foreground, respectively; $p(R_1)$ denotes a shape prior for the foreground. This formulation assumes independence between the foreground and background, and it requires the full knowledge about the complex appearance models $p(\mathbf{I}(R_1)), p(\mathbf{I}(R_0))$ of the foreground and background. However, learning these models is a very challenging task; for example, no existing generative models are able to deal with the appearance and shape variations of the foreground and background in Fig. 1. We therefore choose an alternative perspective which attempts to directly approximate the posterior probability distribution $p(W|\mathbf{I})$.

2.2. An Alternative Perspective

We can express the log posterior distribution $-\log p(W|\mathbf{I})$ as an energy function $E(C; \mathbf{I})$, where C denotes the boundary of the foreground.

We define the energy function by

$$E(C; \mathbf{I}) = E_{dis}(C; \mathbf{I}) + E_{shape}(C) \tag{1}$$

where $E_{dis}(C; \mathbf{I})$ models the appearance cues, and $E_{shape}(C)$ defines a shape prior. Our low- and mid-level models correspond to learning increasingly complex models for $E_{dis}(C; \mathbf{I})$ and the high-level model corresponds to $E_{shape}(C)$.

We define $E_{dis}(C; \mathbf{I})$ by generalizing a pseudo-likelihood function:

$$\begin{aligned} E_{dis}(C; \mathbf{I}) &= -\sum_{\mathbf{s} \in \Lambda / C} \log p(\mathbf{I}(\mathbf{s}), y(\mathbf{s}) = 0 | \mathbf{I}(N(\mathbf{s})/\mathbf{s})) \\ &\quad -\sum_{\mathbf{s} \in C} \log p(\mathbf{I}(\mathbf{s}), y(\mathbf{s}) = 1 | \mathbf{I}(N(\mathbf{s})/\mathbf{s})) \end{aligned} \tag{2}$$

where $\mathbf{I}(\cdot)$ is the intensity value(s) at the given pixels(s); $N(\mathbf{s})$ is a neighborhood on pixel s, $N(\mathbf{s})/\mathbf{s}$ includes all the pixels in the neighborhood except s; $p(\mathbf{I}(\mathbf{s}), y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})/\mathbf{s}))$ is a conditional joint probability. $y(\mathbf{s}) = 1$ indicates that pixel s is on the object boundary.

We can re-express the likelihood function as follows:

$$\begin{aligned} E_{dis}(C; \mathbf{I}) &= -\sum_{\mathbf{s} \in \Lambda} \log p(\mathbf{I}(\mathbf{s}), y(\mathbf{s}) = 0 | \mathbf{I}(N(\mathbf{s})/\mathbf{s})) \\ &\quad -\sum_{\mathbf{s} \in C} \log \frac{p(y(\mathbf{s}) = 1 | \mathbf{I}(N(\mathbf{s})))}{p(y(\mathbf{s}) = 0 | \mathbf{I}(N(\mathbf{s})))} \end{aligned} \tag{3}$$

which can be done by adding

$$-\sum_{\mathbf{s} \in C} \log p(\mathbf{I}(\mathbf{s}), y(\mathbf{s}) = 0 | \mathbf{I}(N(\mathbf{s})/\mathbf{s}))$$

into the first term in the right side of Eqn. 2 and subtracting it from the second term in the right side of Eqn. 2. The first term in Eqn. 3 does not depend on C and hence can be ignored. We write:

$$E_{dis}(C;\mathbf{I}) = -\sum_{\mathbf{s}\in C} \log \frac{p(y(\mathbf{s})=1|\mathbf{I}(N(\mathbf{s})))}{p(y(\mathbf{s})=0|\mathbf{I}(N(\mathbf{s})))}, \quad (4)$$

where $p(y(\mathbf{s})=1|\mathbf{I}(N(\mathbf{s})))$ is the posterior probability of a pixel $\mathbf{s}$ belonging to the object boundary given a neighborhood centered at $\mathbf{s}$.

The next section discusses how we learn $E_{dis}(C;\mathbf{I})$ to model low- and mid-level cues. Ideally, we would also learn the shape prior E_{shape} but this is beyond the scope of this paper. Instead we build on previous work [19] and use a simple mixture model to define a shape prior by

$$p(C) = \frac{1}{|DB|} \sum_{C_i \in DB} p(C_i), \quad (5)$$

where DB includes all the shape templates manually labeled for the 100 training images, and $p(C_i)$ allows global affine and local non-rigid transformations for a template C_i.

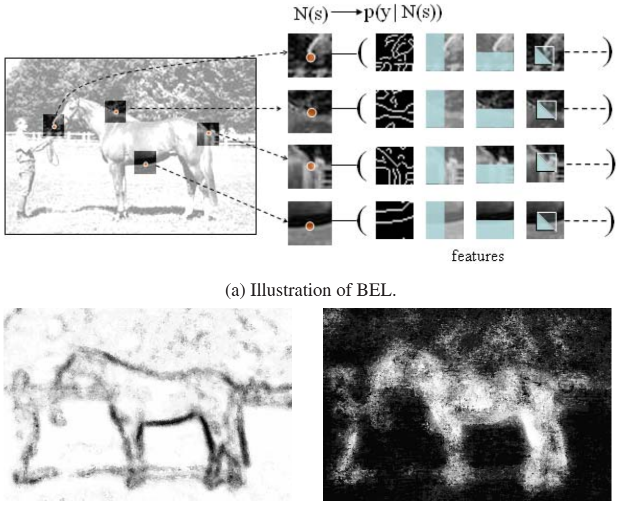
3. Learning $E_{dis}(C;\mathbf{I})$

A key step in our approach is how to learn and compute $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$ in Eqn. 4. As we can see in Eqn. 4, the decision on whether a pixel $\mathbf{s}$ is on the object boundary is decided by a neighborhood of $\mathbf{s}$. There are two extreme situations: (1) the neighborhood is so small that it only contains pixel $\mathbf{s}$. In this case, $E_{dis}(C;\mathbf{I})$ is purely computed by a classifier on a single pixel intensity. Although it is very easy to learn and compute such a classifier, the results are poor because a single pixel intensity is rarely enough to classify a pixel. (2) The neighborhood is as big as the entire image. In this case, the information contained is very sufficient, but learning an accurate $E_{dis}(C;\mathbf{I})$ is extremely difficult.

In this paper, we define cues to be low- and middle-level based on the size of the neighborhood $N(\mathbf{s})$ and the types of features used to learn the distribution $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$. Low-level cues depend on small sized regions while mid-level cues are allowed to have a larger context.

3.1. Low-level Approaches

Classic edge detectors [3, 11] use only gradient information which corresponds to selecting a very small neighborhood $N(\mathbf{s})$ and a restricted class of distributions $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$. The relative ineffectiveness of Canny edge detectors for this task, see Fig. 1, suggests that these neighborhoods are too small, though better results might be obtained by using slightly bigger neighborhoods and explicit learning [9].



(a) Illustration of BEL.

(b) A boundary map by BEL.

(c) Classified foreground.

Figure 2. (a) Illustration of the BEL edge detector; (b) shows a result by the BEL detector; (c) displays a result by a similar classifier learned on the foreground regions.

3.2. Boosted Edge Learning [6]

To model low-level cues, we use a 31×31 neighborhood and learn a classifier $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$. We need a classifier which is easy to learn and fast to compute, therefore we adopt the recently proposed edge detector, Boosted Edge Learning (BEL) [6], which is built on top of a probabilistic boosting tree classifier [17]. Learning and computing an approximation to $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ is briefly described below and we refer to [6] for more details.

For training:

1. Collect a set of training images in which the object boundaries are manually labelled;
2. Sample a number of positive (image patches with a boundary pixel in the center) and negative (image patches with a non-boundary pixel in the center) examples as training set;
3. Train a boosting classifier [7] based on around 50,000 features computed in each image patch, including Canny edge results at different scales, Haar filter responses, gradients, and curvatures;
4. Divide the training set (or bootstrap more samples from the training images) into left and right branches and recursively train sub-trees;
5. Go back to step 3 until a certain stopping criterion is met.

For testing:

1. Scan through the input image pixel by pixel;
2. Compute the discriminative model $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ based on the features selected by the overall classifier from a 31×31 image patch;
3. Output the edge probability map.

In the testing procedure, the overall discriminative probability is approximated by:

$$p(y|\mathbf{I}(N'(\mathbf{s}))) \approx \sum_{l_1} \hat{q}(y|l_1, \mathbf{I}(N'(\mathbf{s})))q(l_1|\mathbf{I}(N'(\mathbf{s})))$$

$$\approx \sum_{l_1,l_2} \hat{q}(y|l_2, l_1, \mathbf{I}(N'(\mathbf{s})))q(l_2|l_1, \mathbf{I}(N'(\mathbf{s})))q(l_1|\mathbf{I}(N'(\mathbf{s})))$$

$$\approx \sum_{l_1,\cdots,l_n} \hat{q}(y|l_n, \cdots, l_1) \cdots q(l_2|l_1, \mathbf{I}(N'(\mathbf{s})))q(l_1|\mathbf{I}(N'(\mathbf{s})))$$

where l_i's are the tree layers in PBT [17], and $q(l_2|l_1, \mathbf{I}(N(\mathbf{s})))$ etc. compute the discriminative model by each AdaBoost node in the PBT.

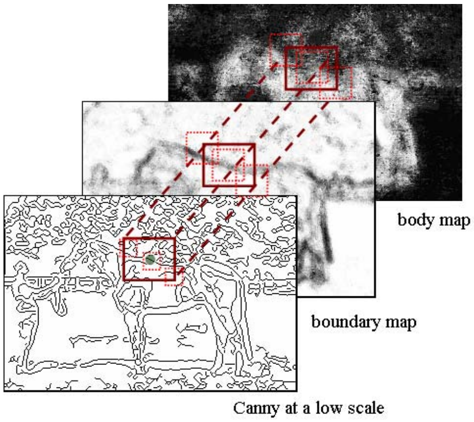
Fig. 2(a) shows an illustration of the BEL detector. Even better results might be obtained using a larger neighborhood, say 61×61, but we lack sufficient data for neighborhoods of this size and, more importantly, it is not clear what features should be used to learn the probability distributions. Instead, we proceed to a compositional approach where we learn distributions for 31×31 neighborhoods and then combine them to get probability distributions on larger neighborhoods.

To assist this compositional approach, we apply the same method to learn the "body map" of the foreground object. The only difference is that a PBT classifier on the pixels on and off the objects, $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ looks at the "difference" between the object boundary and non-boundary pixels, while $p(z(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ classifies the object body itself to account for the texture and lighting patterns implicitly, where $z(\mathbf{s}) = 1$ denotes that pixel $\mathbf{s}$ is on the object body. Fig. 2(b) and Fig. 2(c) show examples of probability map on object boundary and body, respectively. As we can see, the result is better than that shown in Fig. 1.

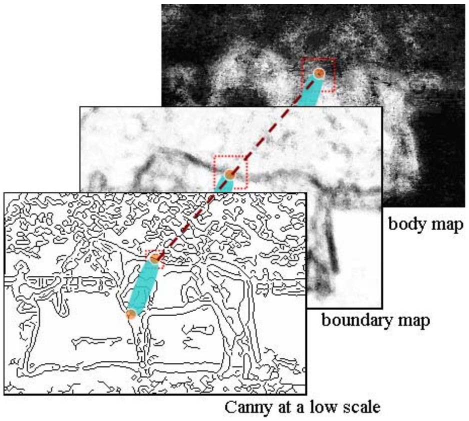
We now proceed to build mid-level classifiers by using PBT to combine features including the PBT boundary map and PBT body map. This combination adds context information by increasing the neighborhood size. Intuitively, the body map in the expanded neighborhood of the edge can support the local evidence for the edge given by the PBT boundary map. For example, the body map should be large on one side of the boundary and small on the other side. This is described in the next section.

3.3. Combing More Context Information

As mentioned above, boundary map $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ and body map $p(z(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ partially capture information for $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$. We want to collectively learn/compute a model for $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$ based on $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$ and $p(z(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$. Although it is possible to combine this information using methods like conditional random fields [10], this leads to time-consuming inference computations requiring either Gibbs sampling [10] or belief propagation [20]. Instead, we learn another classifier to combine this information, which greatly reduces the time for inference. We design two schemes to compute $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$.



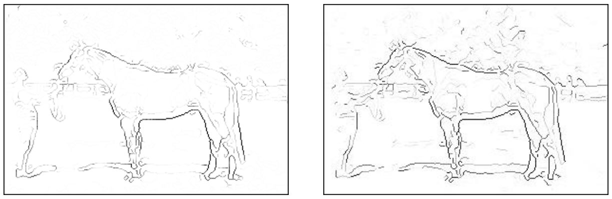
(a) Short-range context information



(b) Long-range context information

Figure 3. Illustration of the second stage classifier accounting for more middle level context information.

Short-range Context Information



(a) Short Range Context.

(b) Long Range Context.

Figure 4. Results by the second stage classifiers on an example.

A straightforward approach is to learn another classifier directly on top of the boundary map and the body map. We call this "short-range" context information. To reduce the number of pixels to be checked, we explicitly use the result by Canny edge detector at a small scale $\sigma = 1.0$ and only those pixels on the Canny edge map will be considered. We still use a neighborhood of size 31×31 centered at pixel $\mathbf{s}$. Note this neighborhood essentially covers the original image patch of 61×61 as each pixel in the boundary map itself is decided by its 31×31 neighborhood. Fig. 3(a) shows an illustration. We train a PBT classifier on $5,000$ features using the three channels. The training/testing strategy is nearly identical to that stated in learning/computing $p(y(\mathbf{s})|\mathbf{I}(N'(\mathbf{s})))$. Fig. 4(a) illustrates the result by this approach on an example. Our results on the testing images show a big improvement over that by BEL in the first layer.

Long-range Context Information

The above classifier accounts for somewhat "short-range" context information. We also design another strategy to account for "long-range" context information, which more explicitly carries middle level information. The basic idea can be briefly described as the following. For any point on the boundary of an object, the point on "the other side" of the object shows some affinity with this point. For a boundary point, we shoot a ray along its normal direction until it hits another boundary point. Often: (1) the intensity patterns between the two points observe some regularity; (2) the local geometric properties on the two points also show some consistency (e.g., they are more or less parallel to each other). Previous approaches often use Gestalt law to account for this type of information by some specifically defined rules. Not only do they need to involve heavy manual design, but an optimal way to combine them is usually not available. Here, we adopt the PBT classifier and use a candidate pool of around $20,000$ features, which include difference between the texture patterns, geometric properties of the two end points, as well as texture property of the images between the two points. Fig. 3(b) shows an illustration. The training process is similar to that in BEL while now we are looking at an elongated bar. In the testing stage, for any boundary point on the Canny edge map, we shoot a ray along its normal direction. For any edge point on the ray, we apply the learned classifier to compute how likely this pair is on the foreground object. Our results also demonstrate an improvement over that by BEL. Fig. 4(b) illustrates the result by this approach on an example.

4. Performing shape matching

Once $p(y(\mathbf{s})|\mathbf{I}(N(\mathbf{s})))$ has been learned, we then need to infer a solution C from Eqn. 1 taking into account the shape prior $E_{shape}(C)$. This task can be directly performed by shape matching. We adopt a scheme reported in [19] which can be viewed as a probabilistic version of shape context approach [1].

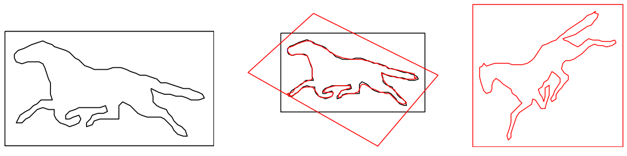

(a) Target shape (b) Transformation (c) Source shape

Figure 5. Illustration of a shape matching case in which a source shape C_i is matched with a target shape C through a transformation function F.

We formulate the E_{shape} term by

$$E_{shape}(C) = E_{matching}(C, F(C_i)) + E_{prior}(F) \quad (6)$$

where C_i denotes one of the templates in the training set, and F accounts for the global affine transformation and local non-rigid deformation on C_i. Intuitively, we want to find the template which best matches with C while not undergoing large deformation. More details can be found in [19, 1] and Fig. 5 illustrates the basic idea.

5. Outline of the Algorithm

Equipped with all the methods in our approach, we are now ready to give an outline of the algorithm.

Training

1. Collect a set of training images in which the object boundaries are manually annotated. We also obtain the corresponding label maps and shape templates.
2. Train a classifier on the object boundaries.
3. Train a classifier on the object label maps.
4. Train another classifier on top of the results by the above classifiers (either the short-range or the long-range method described before).

Testing

1. Run the boundary classifier to obtain an object boundary map.
2. Run the body map classifier to obtain an object body map.
3. Run the context information classifier to obtain a refined boundary map (short-range context classifier gives a better result).
4. Sample a set of points based on the probability map computed in step 3.
5. Use the shape matching algorithm to obtain the best match and output the result.

Fig. 6 illustrates our approach as realizing different levels of information, from low to high. The Canny edge detector accounts for only low-level information as it looks at gradient information only. The BEL edge detector explores both low level and somewhat middle level information. The short-range context information contains more context from different channels while the long-range context information looks at the middle-level (e.g. Gestalt law) information more explicitly. Finally, the high level information steps in to clarify some ambiguity which can not be resolved from the previous steps without explicit shape information.

6. Experiments

Our system was trained on 100 randomly selected images from a dataset consisting 328 horse images [2]. The dataset also contains manually annotated ground truth boundaries. We use the remaining images for testing and we work on gray scale images.

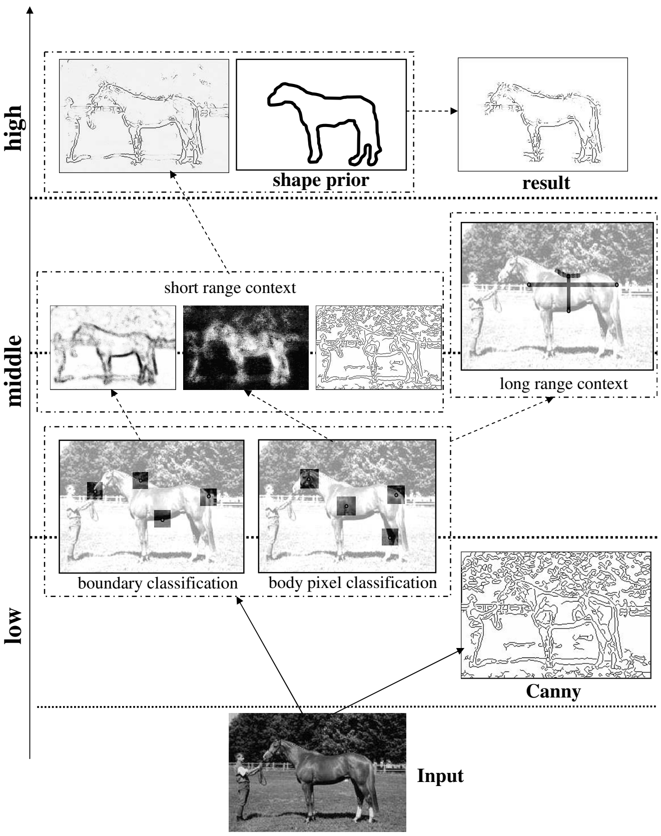


Figure 6. Illustration of our methods engaging the low, middle, and high level information.

Section 5 gives an outline of our algorithm. First, we adopt the BEL edge detector [6] for detecting object boundaries. The training and testing procedures are briefly described in section 3.2. Similarly, we can train a classifier to classify pixels on the foreground regions, which are horse bodies in this paper. We use the same procedures as those for training/testing the boundaries, with different positive and negative examples.

Both the detectors share the same set of candidate features in training, e.g. different Haar filters with different sizes at various locations [22], and histogram features on various Gabor functions. Not surprisingly, the boundary classifier mostly selects Haar filters and the body detector picks more histogram features, which are effective in capturing the complex appearances of the foreground. Fig. 8a shows some testing images and we can see that they have large variations in texture, pose, scale, and lighting patterns. Examples of the detected boundaries and the foreground regions are shown in Fig. 8b and Fig. 8c, respectively. The training stage needs about 10 hours on a computer with 2.4GHz CPU and 1.0GB memory and the testing stage only needs about 15 seconds for a typical 300 × 200 gray scale image.

After the first level information (boundaries and foreground regions) is obtained, we train another classifier to account for more middle level information. We can either use that for the "short-range" or the "long-range" context information. The procedure is stated in section 3.3. In practice, the classifier for short-range context information gives slightly better results (see Fig. 7). One possible reason is that the long-range context considers the affinity between two boundary points. Though they have some consistency in terms of normal direction, curvature, and texture, they observe more variations. This makes the classification task more difficult. We use short-range context classifier in this paper, which combines information from Canny edges at a small scale, and both the boundary and body maps. Once trained, the classifier gives refined boundaries on the testing images, see Fig. 8(d). This step gives the biggest performance improvement in our system.

Once refined object boundaries are obtained, we then use high level shape information to further improve the results. Rather than using an average template of all the training examples [14], we represent our shape model by a mixture model based on all the training examples. We sample around 300 points from a refined boundary map. Each template in the 100 training examples is used to match against these points using the method described in [19]. The best match with the lowest energy is then used. The time spent on matching is about one minute although we have 100 templates. Moreover, since we can annotate each segment of template boundaries, e.g. head, back, legs, the matching result also provides identified parts. Therefore, not only can we detect the object boundaries, but also we are able to tell where the body parts are. This shape matching process obtains an approximation to minimizing the energy function in Eqn. 1. Based on the matched results, we can further remove some false alarms and infer some missing parts. Fig. 7 shows a precision-recall curve after this procedure and Fig. 8(e) demonstrates the final results for some testing images and the different parts of the horses are labelled according to the matching results. The last two rows in both the columns Fig. 8(e) show the "failure" examples by our algorithm, which are due to confusing and cluttered backgrounds.

The shape matching results can further be used to eliminate the false alarms from the body map and thus we can get a cleaner probability map. When thresholding the resulting probability body map at 0.1, we get 95% of the foreground and 76% of the background in the manually labelled map coincide with our classification result. This is better than the performance reported in [2, 20].

We evaluated our system using precision-recall rates; see the curves in Fig. 7. Compared to [14], our system achieves better performance with even fewer training images and more testing images. There are other works [2, 20, 12] in the literature which tackled the same problem. However,

[20] and [12] worked on color images while we only used gray scale images. Moreover, the details of the performance evaluation in [2, 20, 12] are not so clear. The speed of our algorithm is about 1.5 minutes per image, while speed is not reported in all the above works.

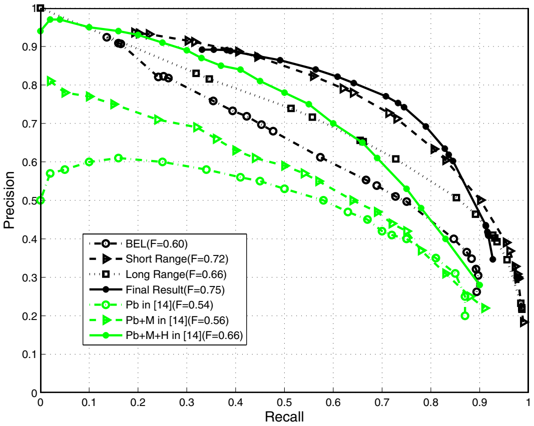


Figure 7. Precision-recall curves by our results on the testing images (three black curves). Results on the same dataset from [14] are also displayed for a comparison (three grey/green curves). The dotted curve shows the result of using long-range context information. The F value shown in the legend is the maximal harmonic mean of precision and recall and provides an overall ranking.

7. Conclusions and Discussions

In this paper, we have proposed a general learning based approach for object boundary detection and segmentation. The classifiers described in this paper implicitly and explicitly account for information including low-level, middle-level and context. Instead of leaving the computing burden to sophisticated inference algorithms [18, 20, 12], our algorithm couples modelling and computing at its early stage. Therefore, the computing processes described in this paper are straight forward and directly tied to the training processes. It takes around 1.5 minutes on a modern PC to segment a horse from an image of size 300 × 200. The learning processes avoid heavy manual design and the very same training program can be used to tackle a variety of classes of objects using an identical setting of parameters. We have shown significant improvement over existing approaches with detailed precision-recall curves. Moreover, the effectiveness of each step of our algorithm is clearly clarified in the error measurement. This facilitates future research of bringing in other components and identifying what role of each level of information plays.

The long-range context information seems to be a bit worse than short-range context. This deserves further study as the former naturally carries richer information than the latter. Also, we will apply our system to other datasets with different objects of interest. More thorough experiments will also be conducted to test the robustness of our algorithm on images of different sizes, cluttered conditions, and multiple conditions.

Acknowledgment S.Z. and Z.T. were funded by the NIH, Grant U54 RR021813 entitled Center for Computational Biology (CCB). We thank the anonymous reviewers for giving many constructive suggestions in carrying out thorough experimental evaluations.

References

[1] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts", *PAMI*, vol. 24, no. 24, 2002.

[2] E. Borenstein, E. Sharon and S. Ullman, "Combining top-down and bottom-up segmentation", *Proc. IEEE workshop on Perc. Org. in Com. Vis.*, June 2004

[3] J. F. Canny, "A Computational Approach to Edge Detection", *PAMI*, Nov. 1986.

[4] S. Dickinson, A. Pentland, and A. Rosenfeld, "From Volumes to Views: An Approach to 3-D Object Recognition", *Comp. Vis. and Image Und.*, Vol. 55, No. 2, March 1992.

[5] B. Dubuc and S.W. Zucker, "Complexity, Confusion, and Perceptual Grouping", *IJCV*, 2001.

[6] P. Dollár, Z. Tu, and S. Belongie, "Supervised learning of edges and object boundaries", *Proc. of CVPR*, 2005.

[7] Y. Freund and R. Schapire, "A Decision-theoretic Generalization of On-line Learning And an Application to Boosting", *ICML*, 1996.

[8] R. Fergus, P. Perona, and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning", *Proc. of CVPR*, 2003.

[9] S. Konishi, A. L. Yuille, J. M. Coughlan, and S. C. Zhu. Statistical edge detection: Learning and evaluating edge cues. *PAMI*, 25(1):57–74, Jan. 2003.

[10] S. Kumar and M. Hebert, "Discriminative random fields: a discriminative framework for contextual interaction in classification", *Proc. of ICCV*, 2003.

[11] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active Contour Models," *Int'l J. Computer Vision*, vol. 1, no. 4, pp. 321-332, Jan. 1988.

[12] M. P. Kumar, P.H.S. Torr, and A. Zisserman, "OBJCUT", *Proc. of CVPR*, 2005.

[13] D. Marr. *Vision*. W.H. Freeman and Co. San Francisco, 1982.

[14] X. Ren, C. Fowlkes, and J. Malik, "Cue integration in figure/ground labeling", *Proc. of NIPS*, 2005.

[15] C. Taylor and D. Kriegman, "Structure and Motion from Line Segments in Multiple Images", *PAMI*, 1995.

[16] A. Torralba, K. P. Murphy and W. T. Freeman, "Contextual Models for Object Detection using Boosted Random Fields", *Proc. of NIPS*, 2005.

[17] Z. Tu, "Probabilistic boosting-tree: learning discriminative models for classification, recognition, and clustering", *Proc. of ICCV*, 205.

[18] Z. Tu, X. Chen, A. Yuille, and S.C. Zhu, "Image parsing: unifying segmentation, detection, and object recognition", *IJCV*, 2005.

[19] Z. Tu and A. Yuille, "Shape Matching and Recognition–Using Generative Models and Informative Features", *Proc. of ECCV*, May 2004.

[20] A. Levin and Y. Weiss, "Learning to combine bottom-up and top-down Segmentation", *Proc. of ECCV*, 2006.

[21] S. Ullman R. Basri "Recognition by Linear Combinations of Models", *PAMI*, May, 1991.

[22] P. Viola and M. Jones, "Fast Multi-view Face Detection", *Proc. of CVPR*, 2001.

[23] D. H. Wolpert, "Stacked generalization", Neural Networks 5(2): 241-259, 1992.

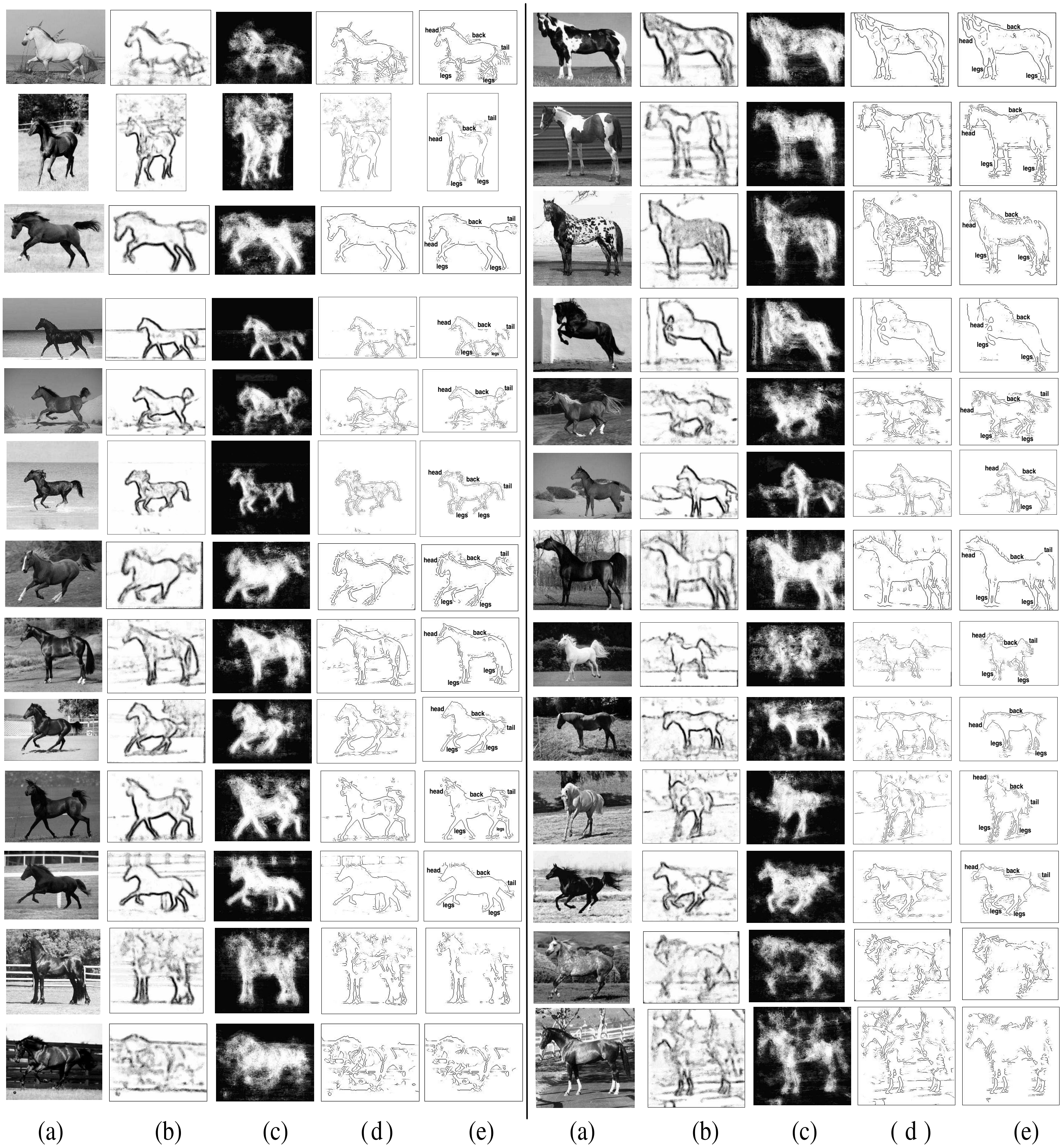


Figure 8. Results on some testing images. (a) shows input images in gray scale. (b) are the boundary maps detected by BEL. (c) shows the body maps. (d) demonstrates refined boundaries based on the short-range context information. (e) gives final results after shape matching, and also the different parts of the horses are labelled according to the shape matching results. These images are representatives of the data set, which have different appearances, poses, scales, and lightening conditions. We see how each level of information is helping detecting their boundaries. The last two rows in both the columns show two worst results by our algorithm.