Recognition of Box-like Objects by Fusing Cues of Shape and Edges

Chia-Chih Chen and J. K. Aggarwal
The University of Texas at Austin, Department of Electrical and Computer Engineering
Austin, TX 78712-0240, USA
{ccchen | aggarwaljk}@mail.utexas.edu

Abstract

Boxes are the universal choice for packing, storage, and transportation. In this paper we propose a template-based algorithm for recognition of box-like objects, which is invariant to scale, rotation and translation as well as robust to patterned surfaces and moderate occlusions. The algorithm first over-segments the input image to partition objects into pieces. Based on the smoothness property of surface texture, candidates for component segments of boxes are selected. Guided by a template trained Linear Discriminant Analysis (LDA) classifier, box-like segments are reassembled from these segments of interests. For each box-like segment, we estimate its probability of being a 2D projection of a 3D box model upon the extracted contour and inner edges. Experimental results demonstrate high detection accuracy of boxes and reliable recovery of their 2D models.

1. Introduction

The monitoring, identifying, (un)loading, and counting numbers of boxes (Figure 1(a)) are of serious importance in industrial and other applications. The success of these tasks largely relies on the results of localizing and modeling available cues of boxes. Our algorithm extracts then verifies cues from the reassembled box-like segments in a color image. According to Pohlke's theorem [3], any three non-collinear line segments in a plane with the same endpoint can be considered as the parallel projection of a 3D cube. Therefore, if this kind of line segment arrangement can be found from the 2D projection of a 3D object, we can infer that the 3D object is box-like. However, for a box-like object to reveal the above mentioned line segment structure at the projection of surface intersections, at least two of its component surfaces must be observable or can be reconstructed (our assumption).

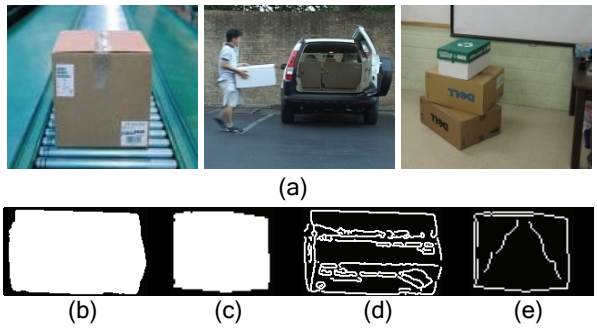


Figure 1: (a) Applications relying on the recognition of boxes (b),(c) box-like binary blobs of segments (d),(e) corresponding edge maps of segments in (b),(c).

Recent works related to this problem include model recovery [5], detection [6,8], and dimension estimation [4]. Katsoulas [5] worked on model recovery of piled box-like objects. His work employs range imagery to avoid the interference of illumination variation and textured surfaces. He simplifies the problem with the additional cue of depth at the greater cost of using a laser range device. Newcorn [6] and Weinand [8] dealt with box detection using intensity images from multiple views. However, their methods are not robust to illumination changes, and are based on the assumption that the objects are properly placed. Fernandes et al. [4] focused on computing the dimensions of a box from single perspective images. Their technique assumes the availability of background and requires a customized laser device.

In this investigation, boxes are assumed to be rigid objects with straight, parallel edges that make right angles at each surface. Other than geometric structure, there is no distinctive feature that may be used to reliably characterize any box-like objects. From the binary blob in Figure 1(b), we can roughly consider it as the shape of a projected box. However, with the additional cue of its edge image 1(d), we are able to confirm our judgment. Similarly, we can easily reject the shape in 1(c) as a box projection by referring to the edge map in 1(e). In this paper, we combine the cues of

contour and inner edges to verify candidate box segments with templates projected from 3D rendered boxes. Owing to the use of modified chamfer [1] distance, the result of model recovery is robust against patterned box surfaces and loss of structural edges.

2. Template-based recognition algorithm

Given an input image, our algorithm aims at detecting and fitting models to the image segments that are likely to be 2D projections of 3D box-like objects. We adopt a bottom-up approach, which is composed of three steps as follows.

2.1. Extraction of seed segments

Even with the optimal parameter setup, there is no guarantee a single image segmentation technique can segment out the complete contours of box-like objects in a given image. Therefore, we over-segment the input image to acquire the constituent segments of boxes. The idea is to reassemble the original object from the pieces in the later processing. There are many image segmentation algorithms that may suit this purpose. However, to preserve the discontinuous boundaries between objects, mean shift based image segmentation algorithm [2] is adopted. Mean shift performs segmentation based only on local image information; as a result, it is able to output segment boundaries that delineate well the true objects.

Using the smoothness property of the texture on box surfaces, we can limit the number of segments to be considered as candidate components of boxes. Here, segment means the binary blob of an image segment. By iteratively merging a candidate segment with its adjacent segments, there may be one combination of segments that reassembles a complete box blob. We call these candidate segments seed segments. To localize the seed segments, the entropy of texture for each grayscale image segment is calculated. The seed segments are defined as segments whose entropy values are among the lowest α% of all segments. The value of α should be large enough such that at least one component segment of every box is selected as a seed.

2.2. Mergence of the box component segments

In this step of the algorithm, various combinations of the seed segments and their adjacent segments are merged and examined by a pre-trained shape classifier. As shown in Figure 2(a), the original image is segmented by mean shift algorithm. After thresholding on entropy of segment texture, the seed segments are extracted. The black segment in 2(b) is one seed

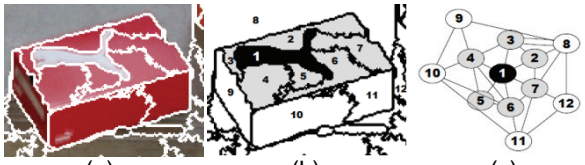


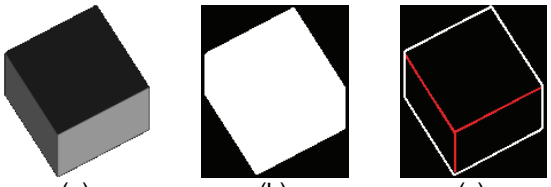
Figure 2: (a) the original image overlaid with segment boundaries (b) the seed segment (black), its directly connected neighbors (gray), and indirectly connected neighbors (white) (c) segment graph: originated from the seed, nodes of segments are organized by an undirected graph.

segment, which is dilated to associate with its directly connected segments (gray segments). In the same manner, the indirectly connected segments (white) can be linked to the directly connected ones. The geometric relations between the seed segment and its surrounding segments can be described by an undirected graph (segment graph) in 2(c). After a sufficient number of iterations (4 in our implementation), the segment graph of each box component will be expanded to cover the set of segments that composes the source box blob. The seed segment {1} in 2(b), for example, is one component segment of the box in 2(a). The nodes in graph 2(c) represent all segments that can be reached by the seed segment within 2 iterations of spatial association. We find that the set of box member segments, {1, 2, …, 11}-{8}, is actually a subset of the nodes in 2(c).

In order to retrieve the subsets of segments that may possibly reassemble a complete box, we trim unlikely sets from all combinations of graph nodes. The candidate sets of a box must follow three basic rules: Seed segment is an element of the set. The merged blob of the corresponding set is a single connected component. The size of the merged blob is smaller than an area threshold. After pruning, we combine segments according to the surviving sets and extract shape features from each merged blobs. The feature vector used to characterize a merged segment contains the following properties:

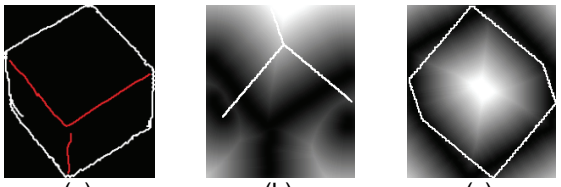
- *Solidity* - the extent to which the blob covers its convex hull area
- *Extent* - the proportion of the blob area to the bounding box area
- *Eccentricity* - the ratio of major axis to minor axis of the ellipse that envelopes the blob
- *Compactness* - the ratio of the blob area to squared perimeter of the circle that envelopes the blob (multiplied by 4π for normalization)

Note that these shape properties are scalar quantities ranging from 0 to 1. In the training phase, the same feature vector is extracted from every 2D box contour template. Figure 3(b) shows a box contour template,



(a) (b) (c)

Figure 3: From a three-surface box model (a), we extract one shape template (b) and two edge templates (c). The edge templates include the inner edge template (red) and the outer edge template (white).



(a) (b) (c)

Figure 4: (a) a merged box-like segment (b) the placement of the transformed inner edge template on the segment inner edge distance map (c) the same arrangement as (b) for the outer edge template and distance map.

which is obtained from the projection of a 3D box model in 3(a). To further reduce the number of segments to be considered, we train a LDA classifier using all contour templates as samples. The merged segments derived from the same seed segment are classified by the LDA classifier. Based on Pohlke's theorem, the template trained classifier divides each segment into one of the three classes, which are two-surface box, three-surface box, or non-box. For all the derived segments of a seed segment, we preserve only segments whose shapes are most similar to the possible plane projections of a box. This is done by locating segments with the highest discriminant value in either of the two box categories. In other words, at most two segments and their corresponding edge maps per seed are passed to the next stage for verification.

2.3. Verification of box-like segments

The salient idea of the proposed algorithm is to verify the shapes and edges of the merged box-like segments with the corresponding types of templates. We formulate the process of template matching into an optimization problem, and propose a combined metric to evaluate the similarity between a candidate segment and a transformed template.

2.3.1. Model generation. To generate templates, we rendered five 3D box models with common specifications. Figure 3(a) shows a plane projection of one of the 3D box models, from which we can acquire one three-surface and three two-surface 2D box surface models. Each box surface model provides one template feature set, $\boldsymbol{T} = <T^S, T^{Ein}, T^{Eout}>$, which corresponds to the triplet of shape, inner edge, and outer edge template. The same of feature set ***Seg*** can be obtained from each merged segment, where ***Seg*** = $<Seg^S, Seg^{Ein}, Seg^{Eout}>$.

The algorithm matches a merged segment with only top similar templates in shapes. As introduced in section 2.2, a shape feature vector **v** is extracted from the merged segments and every shape template. The shape dissimilarity d_S between the segment Seg_i^S and the shape template T_j^S is defined as:

$$d_S(Seg_i^S, T_j^S) = \frac{\|\mathbf{v}_{i,j}\|}{2} + \sigma(\mathbf{v}_{i,j}) \qquad (1)$$

where $\mathbf{v}_{i,j}$ is the difference vector of $\mathbf{v}(Seg_i^S)$ and $\mathbf{v}(T_j^S)$. σ denotes standard deviation. For each Seg_i^S we calculate the shape dissimilarity to every the shape template, and select $N = N_{w/i} + N_{w/o}$ template sets to perform joint feature verification. Here $N_{w/i}$ is the number of template sets with the lowest d_s chosen from the LDA suggested box class, and $N_{w/o}$ is the number of most similar sets picked up from the other box category. Templates from both box categories are included to safeguard the decision of the classifier.

2.3.2. Probability estimation. A probability estimate (2) determines how likely a merged segment feature set $\boldsymbol{Seg_i}$ is the transformed projection of a 3D box model given cues of segment shape and edges separately. Among N candidate template sets, we seek a member template feature set $\boldsymbol{T_j}$ and a specific transformation $tf(\theta)$ that maximize:

$$p(\boldsymbol{Seg}_i = tf(\boldsymbol{T}_j;\theta) \mid Seg_i^S, < Seg_i^{Ein}, Seg_i^{Eout} >) \qquad (2)$$

where θ describes the parameters that control the rotation and scale of the template set. When matching $\boldsymbol{T_j}$ with $\boldsymbol{Seg_i}$, within a fixed number of transformations, there is a unique θ that maximizes (2). In other words, for a specific $\boldsymbol{Seg_i}$, θ is a dependent variable of template number j. Therefore, considering (2) as our objective function, the process of verification is simplified to solving a single-variable (j) optimization problem. Note that both segment contour (Seg^S) and segment outer edge (Seg^{Eout}) delineate the shape of a segment. By treating them as the same feature, our problem becomes:

$$\max_j p(\boldsymbol{Seg}_i = tf(\boldsymbol{T}_j;\theta(i,j)) \mid Seg_i^{Ein}, Seg_i^{Eout}) \qquad (3)$$

From the solution of (3), we are able to estimate how well the segment feature set $\boldsymbol{Seg_i}$ can be modeled by top similar template sets. The argument of maximum j^* indicates the box surface model that matches the segment best, and how the model should

be transformed ($\theta(i, j^*)$) to position on the original image. The evaluation of (3) involves combining similarities between the corresponding edge features (inner and outer edges). When discovering edges from a grayscale box segment, it is almost inevitable to find edges other than those from the intersections of box surfaces. We need a shape matching algorithm that is both robust against cluttered background edges and is computationally efficient. According to [7], chamfer distance suits both requirements well.

For one type of transformed edge template $tf(T^E)$, the computation of chamfer distance from which to the same type of segment edge Seg^E involves transforming Seg^E into a distance map $DT(Seg^E)$. The value of each pixel in $DT(Seg^E)$ represents the Euclidean distance from the pixel to the nearest edge feature in Seg^E. After overlaying $tf(T^E)$ onto Seg^E, the chamfer distance from $tf(T^E)$ to Seg^E is calculated by averaging the values of pixels in $DT(Seg^E)$, whose coordinates are under the edge features of $tf(T^E)$.

However, regular chamfer distance does not fit our application due to two main reasons. First, the distance is not a normalized value. Second, chamfer distance considers the nearest feature to a pixel as a match even if the true correspondence does not exist. Therefore, we propose a modified chamfer distance (4), which converts Euclidean distance into Gaussian-weighted distance and is robust against missing features.

$$d_{cfm}(T^E, Seg^E) = \frac{\sum_{t \in T^E} e^{-\frac{d_{Seg}(t)}{2\sigma^2}} \cdot (\mathrm{sgn}(d_{Seg}(t) - \tau) + 1)}{2N_t(T^E)} \quad (4)$$

In (4), $d_{Seg}(t)$ is the distance from an edge pixel t in T^E to the nearest feature in Seg^E, and $N_t(T^E)$ is the number of edge pixels in T^E whose d_{Seg} is less than τ. The truncation removes the d_{Seg} of erroneously matched edges to ensure the quality of the calculated distance. It is reasonable to assign the values of τ and σ based on the size of the segment, because we expect higher template alignment errors with larger segments. In our implementation, the values of τ and σ are 1/5 and 1/8 of the segment (bounding box) diagonal length.

From a merged segment Seg_i, we can extract the edge map in Figure 4(a). The grayscale image in 4(b) shows the placement of a $tf(T_j^{Ein})$ on $DT(Seg_i^{Ein})$. The same arrangement for $tf(T_j^{Eout})$ and $DT(Seg_i^{Eout})$ is shown in 4(c). The joint similarity between T_j and Seg_i is calculated by combining the chamfer distances between the two types of features:

$$\max_k \left(\frac{N_t(tf(T_j^{Ein};\theta_k)) \cdot d_{cfm}(tf(T_j^{Ein};\theta_k), Seg_i^{Ein})}{N_t(tf(T_j^{Ein};\theta_k)) + N_t(tf(T_j^{Eout};\theta_k))} + \frac{N_t(tf(T_j^{Eout};\theta_k)) \cdot d_{cfm}(tf(T_j^{Eout};\theta_k), Seg_i^{Eout})}{N_t(tf(T_j^{Ein};\theta_k)) + N_t(tf(T_j^{Eout};\theta_k))} \right), \forall k \quad (5)$$

We use (5) to approximate the objective function in (3), and solve (3) by examining candidate templates.

3. Experimental results

We evaluate our algorithm by showing quantitative as well as qualitative results. The measures employed for evaluation are *precision* and *recall*. These two metrics are widely used when the successful detection of one class is more important than the other. In the context of box recognition, the number of non-box segments is usually much more than that of the box segments.

45 color images captured from 10 different scenes are used to test the algorithm. The contents of these images include a stack of boxes, boxes mixed with other objects, a person carrying a box, and boxes on a moving trailer. There are total 201 objects presented in our testing set, among which 117 objects are boxes. We assign α to 40 for the designation of seed segments. The probability threshold is 0.6, segments with greater probability values (Eq. 3) are accepted as box-like objects. This setup gives us 111 true positives (tp), 9 false positives (fp), and 6 false negatives (fn). Therefore, the precision of our algorithm is 92.50% (tp/(tp+fp)) and the recall is 94.87% (tp/(tp+fn)). The false negatives are mainly due to segmentation errors or occlusions. As shown in Figure 5, in addition to our own testing set, we also tested our algorithm with images collected from Google Image Search.

The complexity of the verification stage (bottleneck of the algorithm) is linear to the number of box-like segments, number of candidate templates, and the resolution of rotation. In MATLAB implementation on a Pentium D 2.8GHz PC, the average time required to process a merged segment is about 25 seconds. The overall time required to process an image containing 2 box segments is about 2 minutes (bias with regard to the total number of seed segments). We can further save the computation by testing few major orientations first, and refine the search accordingly.

4. Conclusions

In this paper, we have presented an algorithm for box-like object recognition in color images. By employing a supervised merging process, the number of box-like segments to be verified is considerably reduced. The template-based joint feature verification has two major advantages. First, it offers a meaningful way to combine similarities between contours and inner edges into a probability estimate. Second, the precision of model recovery is noticeably improved due to the separate matching of different feature pairs (inner and outer edge pairs) and the decision on a fused measure.

Figure 5: Representative results of box detection via model fitting from the testing set and images collected from Google Image Search

The modified chamfer distance provides a reliable similarity measure even when the box structural edges are missing. The framework of our algorithm can be generalized to recognize other kinds of rigid objects by different arrangement of template database.

Acknowledgements

This research was supported in part by Texas Higher Education Coordinating Board award # 003658-0140-2007.

References

[1] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf. Parametric Correspondence and Chamfer Matching: Two New Techniques for Image Matching. *Proc. 5th Int'l Joint Conf. Artificial Intelligence*, 659-663, 1977.

[2] D. Comaniciu, P. Meer. Mean Shift: A Robust Approach toward Feature Space Analysis. *IEEE Trans. Pattern Analysis Machine Intelligence*, 24: 603-619, 2002.

[3] H. Döme. *100 Great Problems of Elementary Mathematics: Their History and Solution*, Dover, NY, 1965.

[4] L. Fernandes, M. Oliveira, R. Silva, G. Crepo. Computing Box Dimensions from Single Perspective Images in Real Time, *Proc. of the Brazilian Sym. on Computer Graphics & Image Proc.*, 155-162, 2005.

[5] D. Katsoulas. Reliable Recovery of Piled Box-Like Objects via Parabolically Deformable Superquadrics. *IEEE Int'l Conf. on Computer Vision*, 2: 931, 2003.

[6] D. Newcorn. Robot Gains Eyesight, *Packaging World*, 1998.

[7] A. Thayananthan, B. Stenger, P.H.S. Torr, R. Cipolla. Shape Context and Chamfer Matching in Cluttered Scenes. *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1: 127-133, 2003.

[8] S. Wienand. Robot Vision for Depalletizing of Mixed Pallet Loads. Machine Vision Online, 2003.