Coarse-to-Fine Object Recognition Using Shock Graphs

Aurelie Bataille and Sven Dickinson

University of Toronto

Abstract. Shock graphs have emerged as a powerful generic 2-D shape representation. However, most approaches typically assume that the silhouette has been correctly segmented. In this paper, we present a framework for shock graph-based object recognition in less contrived scenes. The approach consists of two steps, beginning with the construction of a region adjacency graph pyramid. For a given region, we traverse this scale-space, using a model shock graph hypothesis to guide a region grouping process that strengthens the hypothesis. The result represents the best subset of regions, spanning different scales, that matches a given object model. In the second step, the correspondence between the region and model shock graphs is used to initialize an *active skeleton* that includes a shock graph-based energy term. This allows the skeleton to adapt to the image data while still adhering to a qualitative shape model. Together, the two components provide a coarse-to-fine, model-based segmentation/recognition framework.

1 Introduction

Object recognition is one of the primary goals of computer vision, allowing an image signal to be semantically labelled according to a priori knowledge of objects in the world. Early object recognition work in the 60's and 70's focused on the *categorization* or *generic object recognition* problem, in which exemplar objects, i.e., specific object instances, were matched to coarse, prototypical models designed to be invariant to within-class shape and appearance deformation. Although an admirable goal, the low- and intermediate-level infrastructure did not exist to bridge this representational gap [3, 4], leading to systems tested on contrived scenes under contrived viewing conditions.

Over the next 30 years, in a drive toward the recognition of more realistic objects under more realistic imaging conditions, recognition systems became more exemplar-based, beginning with the CAD-based vision systems of the 80's, then moving toward the appearance-based models of the 90's and the recently popular interest-point models. For the first time, complex objects can be recognized in cluttered scenes under varying illumination. However, since such systems are based on the distinguishing local textures of objects and not their prototypical shape, they are ineffective for generic object recognition.

Different object exemplars belonging to a single class may have different color, texture, exact geometry, and part articulation. But at some coarse level of

L. Brun and M. Vento (Eds.): GbRPR 2005, LNCS 3434, pp. 203–212, 2005.

description, the exemplars in a class have similar shape. It is for this reason that the generic object recognition community has focused primarily on shape as *the* class-defining feature. Moreover, the silhouette has emerged as a popular feature with which to characterize the shape of an object. Unlike extracted contours internal to the object, which may reflect either shape or texture, the occluding contour of the object is guaranteed to reflect only shape information. Since the occluding contour depends on viewpoint, 3-D object recognition systems are view-based, in which each generic object is represented as a set of characteristic silhouettes. Provided that an imaged object's silhouette can be extracted from an image, it is matched to a database of silhouettes grouped by object. The closest matching silhouette defines both the identity of the object as well as its pose (depending on the sampling resolution of the viewing sphere over which the silhouettes are captured).

An exact characterization of a silhouette would be appropriate for exemplar-based object recognition. However, since our goal is generic object recognition, we require a silhouette-based representation that is invariant not only to scale, translation, rotation, and occlusion, but part articulation, within-class shape deformation, and minor rotation in depth. The shock graph [11] has emerged as a powerful, generic shape description possessing these properties, and is based on a labelling and partitioning of the skeleton points (shocks) making up the medial axis transform of a shape. Shocks are labelled according to four qualitatively-defined classes, with contiguous clusters of homogeneously labelled shocks comprising the nodes in a shock graph. In the last 5 years, shock graphs have led to a number of successful silhouette-based recognition systems based on graph matching, e.g., [12, 8, 13, 9, 6, 5].

A careful examination of the shock graph-based recognition literature will show that most, if not all, approaches are typically applied to unoccluded, pre-segmented closed contours, with a only few approaches, e.g., [12, 9], tested on occluded shapes. Clearly, the shock graph-based recognition community has focused more on the shape description and matching problem and less on the segmentation of the shapes. The shock graph community has therefore, and understandably, met with resistance and skepticism from those who claim that since region segmentation is an unsolved problem, and since the occluding contour (silhouette) of an object requires that the object's region be correctly segmented, the whole notion of a shock graph rests on a weak foundation.

One can argue that the space of region segmentation errors is equivalent to the space of possible occlusions, for region over-segmentation can be modelled as an undetectable occluder (yielding a truncated silhouette) and region under-segmentation can be modelled as the union of a detectable occluder and the target object (yielding a silhouette that extends beyond the object). However, even though this argument has been made, e.g., in [12, 10, 6], it has not been made convincingly with extensive simulation of segmentation errors. Until testing is performed on real images of real objects, with massive over- and under-segmentation, the shock graph recognition framework will remain on the fringe of the object recognition community.

Occlusion testing in the shock graph community typically involves subjecting a target shape to minor occlusion. The shock graph is a distributed representation, with nodes corresponding to distinct parts, and one would expect that the occlusion of one part will not affect the representation of another. Although this is indeed true for well-separated parts on an object, occlusion can, in fact, result in major changes in the topological structure of an object's skeleton, yielding major changes in its shock graph structure. Thus, in the presence of significant region over- and under-segmentation, the resulting region's shock graph may bear little resemblance to the model shock graph to which it should match. Since one cannot guarantee that region segmentation errors are minor, we need an approach that couples object recognition using shock graphs with the underlying region segmentation problem, yielding a segmented shape that closely resembles a model object. This paper addresses this problem, and proposes a two-part solution.

2 Region Segmentation and Description

We begin by constructing a region adjacency graph pyramid or scale space, based on varying the parameters of the region segmentation algorithm (Comaniciu and Meer [1]). By varying the segmentation parameters, we can obtain a variety of segmentations, from heavily under-segmented to heavily over-segmented. The resulting regions at a given level may not correspond to objects in the image due to segmentation errors. However, the correct boundary of a given object may, in fact, span multiple scales. Each region segmentation level yields a region adjacency graph, with nodes representing region boundaries and edges specifying region adjacency.[1] Each node (region boundary), in turn, is represented by a shock graph. The region adjacency graphs are linked together to form a tree or pyramid, with a node at a coarser level pointing to one or more component nodes at the next finer level.

3 Model-Based Region Grouping

Given our pyramid of region adjacency graphs, our goal is to try and segment and recognize the object(s) in the scene. Using a model hypothesis for a given region in the image, we will search the space of possible merges of adjacent regions, at different scales, in an effort to strengthen the hypothesis beyond some appropriate threshold. We proceed in a top-down manner, starting with hypotheses for regions at coarser levels before proceeding to region hypotheses at lower levels. By merging adjacent regions at different scales, we consider the space of discrete "outward" perturbations of a region's shape, whereas by descending to a lower level when generating hypotheses, we consider the space of discrete "inward" perturbations of a region's shape. Such perturbations amount to moving

[1] We gratefully acknowledge the region adjacency graph construction module provided by Sven Wachsmuth.

between adjacent levels in the pyramid, and we use model hypotheses, invoked by region shape, to guide the traversal of the pyramid.

The model hypotheses are generated according to the framework described in [6], as shown in Figure 1. For each region at the coarsest level, the region's shock graph is indexed into the database of model shock graphs, returning a small set of promising candidates. These candidates, along with their similarity to the model (computed by a matching algorithm), form the initial "open list", sorted by a cost function, in the traditional graph search algorithm (Nilsson [7]). As shown in the algorithm described in Figure 2, the first element, or state, on the list is removed and tested to see if it's a solution. If not, the state is expanded to yield a set of successor states, in this case the set of possible merges of adjacent regions at the current or finer scales. If any of these successors results in a region whose shock graph is closer to the model than the expanded hypothesis, the successor is merged onto the open list (according to its evaluated cost). Although the algorithm terminates when a solution has been found, it may continue if there are other objects in the scene, i.e., regions not accounted for.

To illustrate the generation of successors, consider the example shown in Figure 3, depicting three levels of segmentation. Consider the red ellipse at the coarsest level. It's successors at that level include its merge with the light blue region to the left, and its merge with the pink region to the right. Its footprint is shown in levels 2 and 3 by the dotted lines. At level 2, its two successors are shown with black outline, while at level 3, its three successors are also shown with black outline. Only those successors that improve the quality of the match between the region's shock graph and the hypothesized model's shock graph, are added to the open list; the rest are discarded. Finally, the cost function used to rank the states on the open list governs the order in which hypotheses are considered. In our experiments, we adopt a "best-first" approach, in which the most promising hypotheses are expanded first, regardless of which levels their component regions are drawn from. We have also explored a "breadth-first" strategy, which favors hypotheses at coarser levels, as well as a "depth-first" strategy, which favors hypotheses at finer levels.

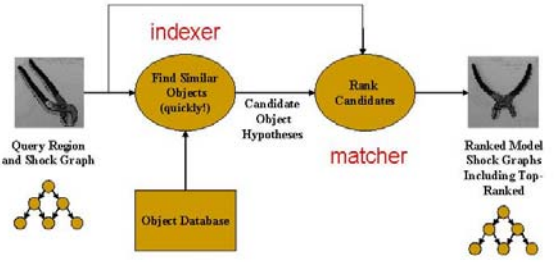

Fig. 1. Generating Model Shock Graph Hypotheses for a Given Image Region Shock Graph

```
for level = 0 to maxLevel do {consider each level, from coarse to fine}
  for all region R of level do {match all regions at the current level}
    OPEN.push(match(R))
  end for
  while OPEN not empty do {if open is empty, go to the next level}
    sort(OPEN, f) {sort open given the cost function selected}
    (R, M) = OPEN.pop() {Consider the first element of the open list}
    CLOSED.push((R, M)) {Move it to the closed list.}
    if (R, M) is a solution then {if a solution is found, exit with success}
      exit((R, M))
    end if
    for lev = level to maxLevel do {if it is not a solution, expand its children at each finer level}
      R' = footPrint(R, lev) {get the footprint of the region}
      ADJREGIONS = adjRegions(R') {get the regions adjacent to it}
      for all region R'' in ADJREGIONS do {try merging each adjacent region with the
      "footprint"}
        R* = merge(R', R'')
        if sim(R*, M) > sim(R, M) then {add the merge only if it improves the similarity with
        the model}
          OPEN.push((R*, M))
        end if
      end for
    end for
  end while
end for
exit("NO_SOLUTION") {if did not exit earlier, no solution was found}
```

Fig. 2. Algorithm for Performing Model-Based Region Grouping

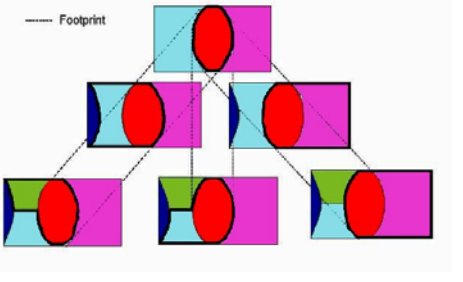

Fig. 3. Successor Generation (see text for explanation)

4 Model-Based Region Fitting

The search space for our model-based merging process is clearly richer than a single region segmentation. However, there may not exist a region segmentation parameter setting (or at least one sampled in the construction of the scale space) that recovers part of an object's boundary. In this case, no amount of model-based merging will recover that part of the boundary. It is here that we use whatever matching contour we have accumulated and return to the image in an attempt to find the contour. Just as we used a shock graph to guide our search through the space of possible region segmentations, we will again use a shock graph to guide our search for the missing contour, in an effort to fine-tune our region to be even closer to the model.

We employ an active contour-like approach, and build in model constraints *based on the shock graph*. Thus, like a traditional active contour approach, the

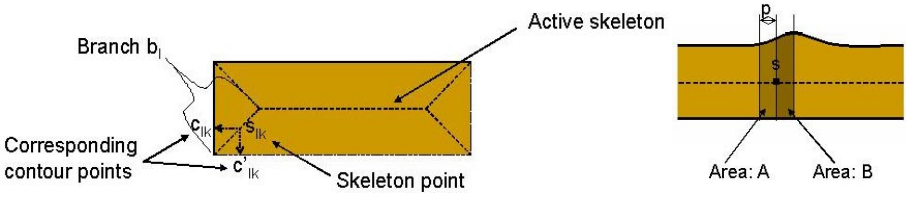

Fig. 4. Region Refinement using an Active Skeleton. Left: active skeleton. Right: the integrated radius function over small windows to the right (B) and left (A) of a skeleton point are used to define the shock graph energy term. For example, in the case of the constant cross-section type 3 node, the energy term would be proportional to the absolute value of the difference in areas

contour is data-driven by gradient structure in the image. However, we diverge from the traditional active contour in two important ways. First, we introduce the concept of an *active skeleton*, to which external (image gradient-based) and internal (smoothness) forces apply. Second, we introduce an energy term to the active skeleton energy minimization that ensures that its shape, while adapting to the image data, conforms to the model shock graph.

A hypothesis emerging from our model-based region segmentation step defines an explicit correspondence between branches (nodes) in the shock graph corresponding to the region group and branches in the model shock graph. Each corresponding branch pair defines a set of corresponding contour points, since each skeleton point defines a pair of contour points, as shown in Figure 4 (left). These matching contour points are used to align the model silhouette to the region group boundary. The same transformation is used to project the model skeleton into the image, representing our initial active skeleton. As image gradient "forces" attract the model silhouette, the positions and/or radii of the active skeleton points are updated to better fit the boundary data. As is common in active contour formulations, the active skeleton is subject to internal smoothness constraints.

Our second departure from traditional active contours is the explicit incorporation of model shape information as a deformation constraint. Since we know the qualitative shape class of a model branch, we can penalize changes in skeleton point position and/or radius that deform the branch shape out of its model class. An example of this additional energy term is shown in Figure 4 (right), in which an energy term, proportional to the slope of the radius function over a window, is used to maintain the branch's type 3 (constant radius) shape. In this way, a qualitative shape model can be folded into an active contour (in this case, skeleton) formulation, providing much stronger deformation constraints.

Our algorithm for region refinement is shown in Figure 5. We loop through each skeleton point on each branch, sampling nearby positions and radii and updating the position and radius of the point if its energy decreases. Once all skeleton points have been visited once, a branch adjustment is performed, allowing an updated branch to "pull" any connected branches in order to maintain the

```
while there is a branch that hasn't converged and MaxIterations have not been exceeded do
  for l in {1, ...., m} do {loop on branches}
    if move[l] then {consider only the branches that haven't converged yet}
      curE = 0
      for k in {1, ...., n} do {loop on skeleton points}
        E_min = infinity
        for all s in U(s_k) do {consider points in s' neighbourhood}
          for all r in R(s) do {radius variation}
            compute the locations of the corresponding contour points c and c' given s and r
            E'(s) = α * E_sm1(s) + β * E_sm2(s) + γ * E'_im(s) + δ * E_shock(s) {compute energy
            at s}
            if E'(s) < E_min then {check if it is the minimum energy; if it is, store the skeleton
            and contour point locations}
              E_min = E'(s)
              s_min = s, c_min = c, c'_min = c'
            end if
          end for
        end for
        curE = curE + E_min
        move s to s_min {move skeleton and contour points to the location that minimizes the
        energy}
        move c to c_min
        move c' to c'_min
      end for
      move skeleton points connected to the branch given branch junction
      if |prevE[l] − curE| < minE then {if the new energy did not improve by much, the branch
      converged}
        move[l] = 0
      else {no convergence yet}
        prevE[l] = curE
      end if
    end if
  end for
end while
```

Fig. 5. Refining the Model using an Active Skeleton

connectivity and integrity of the active skeleton network. Here, we draw on the concept of an active contour network ([2]), in which a set of active contours are connected at junctions using spring forces. The algorithm then iterates, visiting each branch a second time, unless the branch has converged. When all branches have converged, the algorithm terminates.

5 Results

To evaluate the model-based merging procedure, we captured model silhouettes for a variety of views for each of 13 different objects. These objects, as well as different objects belonging to the same set of object classes, were imaged in 15 test scenes, examples of which are shown in Figure 6. Each test scene was region segmented at four levels, resulting in a region segmentation pyramid. Shock graphs were computed for each region and models hypothesized. To evaluate our algorithm, the correctness of the labelled regions (determined from ground truth) was compared to the best results obtained if one were to opportunistically choose from the set of four region segmentations that which yielded the best results without region grouping.

In terms of degree of improvement, in the worst case (best baseline segmentation), 33%, and in the best case (worst baseline segmentation), 45% of the

Fig. 6. Examples of Test Images

hypotheses were incorrect in the baseline system, and became correct under our framework, while there was a 0% change in the other direction. Our system is therefore improving the recognition process considerably. Moreover, among those hypotheses that stayed correct, the majority improved their matching score, with a significant number passing over the solution threshold.

We now demonstrate the model-based region fitting process. Figure 7 shows the four levels of region segmentation used to construct the region adjacency graph pyramid for an input image containing a pair of scissors. Figure 8 illustrates the process of determining that portion of the region group which matches the model and therefore participates in the aligning transformation. Finally, Figure 9 illustrates the initial model aligned in the image, along with the final resting position of the active skeleton. Despite the presence of ambiguous contours in the image, the active skeleton adapts to those contours which preserve the shock graph labels of the individual branches. The classical, somewhat weak active contour formulation is thus strengthened to include a much more flexible shape model that, unlike statistical (active) shape models, needs no extensive training while supporting full articulation and within-class deformation.

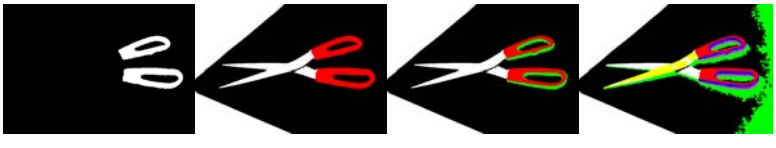

Fig. 7. The four segmentation levels of a test scene containing a scissors

Fig. 8. The points used to compute the model alignment. Left: result of the model-based region grouping, with region group (in green) and matched shock graph (skeleton in blue). Middle: silhouette of model shock graph. Right: those portions (red) of the region group skeleton that match the model shock graph, along with their corresponding contour points (yellow) used to compute the aligning transformation

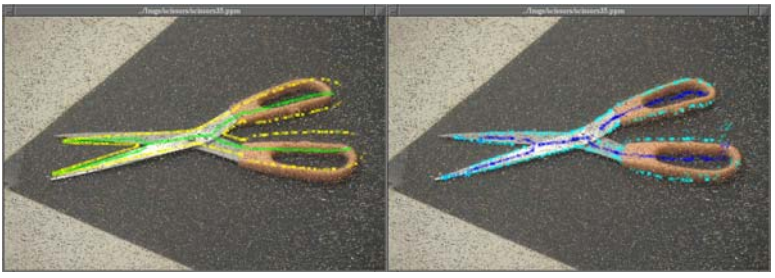

Fig. 9. Model-Based Region Fitting using an Active Skeleton. Left: initialized active skeleton (green) with corresponding contour points (yellow). Right: final active skeleton (dark blue) with corresponding contour points (light blue)

6 Conclusions

Shock graphs offer a powerful framework for representing and matching qualitative shape. Unfortunately, little effort has been devoted to their use in realistic scenes in which a silhouette cannot be properly segmented. In this paper, we attempt to address this problem, drawing on the assumption that since shock graphs do provide locality of representation, portions of regions that are properly segmented provide important clues as to what object is present (i.e., accounts for a particular region). Introducing a region segmentation hierarchy, we can use this model hypothesis to guide a search through a large space of possible splits and merges of the regions. The resulting grouping may, in fact, span many levels of the segmentation hierarchy.

We apply a standard state space search algorithm, and have explored a number of heuristics for ordering the search. In comparing the results to a baseline single region segmentation, we found that our approach often found the correct hypothesis whereas the baseline system did not, and that baseline correct hypotheses improved significantly in our approach. Preliminary results indicate that our multiscale, model-based region grouping framework significantly improves object recognition. Moreover, it is based entirely on a shock graph representation and matching framework, offering hope that shock graphs can be used under more realistic imaging conditions.

The model-based merging framework can be thought of as a mechanism for guiding the search through a discrete space of large-scale perturbations. Unfortunately, there is no guarantee that the correct shape exists in this space, requiring that we return to the image to explore a continuous space of fine-scale perturbations. In the second part of the paper, we again draw on the shock graph, but this time, we use it to constrain an active contour that will settle on the data subject to maintaining a qualitative shape model. We introduce the notion of an active skeleton, an active contour that represents the skeleton and adapts to the image data. Moreover, we add an energy term that keeps the individual skeleton "parts" from deviating from their specified model classes.

References

1. D. Comaniciu and P. Meer. Robust analysis of feature spaces: Color image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 750–755, 1997.
2. S. Dickinson, P. Jasiobedzki, G. Olofsson, and H. Chri stensen. Qualitative tracking of 3-d objects using active contour networks. *Computer Vision and Pattern Recognition*, pages 812–817, June 1994.
3. Y. Keselman and S. Dickinson. Bridging the representation gap between models and exemplars. In *IEEE Computer Society Workshop on Models versus Exemplars in Computer Vision*, Kauai, Hawaii, December 2001.
4. Y. Keselman and S. Dickinson. Generic model abstraction from examples. In *IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, December 2001.
5. S. Kosinov and T. Caelli. Inexact multisubgraph matching using graph eigenspace and clustering models. In *9th International Workshop on Structural and Syntactic Pattern*, 2002.
6. D. Macrini, A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. Zucker. View-based 3-D object recognition using shock graphs. In *Proceedings, Internal Conference on Pattern Recognition*, Quebec City, August 2002.
7. N. J. Nilsson. *Principles of Artificial Intelligence*, chapter 2, Search Strategies for AI Production Systems. Tioga Publishing Co., 1980.
8. M. Pelillo, K. Siddiqi, and S. W. Zucker. Matching hierarchical structures using association graphs. In *European Conference on Computer Vision*, volume 2, pages 3–6, Freiburg, Germany, 1998.
9. T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing shock graphs. In *IEEE International Conference on Computer Vision*, pages 755–762, 2001.
10. A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. Zucker. Indexing using a spectral encoding of topological structure. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 491–497, 1999.
11. K. Siddiqi and B. Kimia. Toward a shock grammar for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1996.
12. K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. In *ICCV*, pages 222–229, 1999.
13. A. Torsello and E. R. Hancock. Computing approximate tree edit-distance using relaxation labelling. In *Workshop on Graph-based Representations in Pattern Recognition*, pages 125–136, 2001.