Skeleton Simplification by Key Points Identification

Gabriel Rojas-Albarracín[1], Carlos A. Carbajal[1],
Antonio Fernández-Caballero[1,2], and María T. López[1,2]

[1] Instituto de Investigación en Informática de Albacete, 02071-Albacete, Spain
[2] Universidad de Castilla-La Mancha, Departamento de Sistemas Informáticos, 02071-Albacete, Spain
caballer@dsi.uclm.es

Abstract. The current skeletonisation algorithms, based on thinning, extract the morphological features of an object in an image but the skeletonized objects are coarsely presented. This paper proposes an algorithm which goes beyond that approach by changing the coarse line segments into perfect "straight" line segments, obtaining points, angles, line segment size and proportions. Our technique is applied in the post-processing phase of the skeleton, which improves it no matter which skeletonisation technique is used, as long as the structure is made with one-pixel width continuous line segments. This proposal is a first step towards human activity recognition through the analysis of human poses represented by their skeletons.

Keywords: Thinning, skeletonisation, image post-processing.

1 Introduction

Pattern recognition has been and continues to be one of the main lines of research in Artificial Intelligence, especially in the areas of Natural Language processing (voice recognition) and Computer Vision (face and human emotion recognition, handwriting recognition, document classification and many more) through biometric parameters. Some of these techniques are based on the principles found in Psychology (visual intelligence), Biology (human anatomical features), Mathematics, Physics and Statistics. From a human perspective, an object can be recognized by looking straight at it or by looking at a simplified image of it. In the field of Artificial Vision, one of the ways to improve the process of object recognition is through the skeletonisation of the image or of the points of interest to be identified in the object, so that from that or those points of interest, recognition can take place. This step reduces the amount of data to be processed, thus reducing the time spent in object recognition.

The algorithm proposed in this paper is a complement to the post-processing phase of the skeletonized image, in which the skeleton is perfected through the elimination of isolated pixels and the substitution into straight line segments. At the same time, they provide points, angles, line segment size and proportions,

J.A. Carrasco-Ochoa et al. (Eds.): MCPR 2010, LNCS 6256, pp. 30–39, 2010.

which are valid and feasible results for image analysis. The algorithm was tested using two well-known skeletonisation algorithms in different images obtained from different sources by applying a pre-processing. This proposal is an initial step towards human activity recognition through the analysis of human poses represented by their skeletons. Aside from this initial section, the rest of the paper is made up of 4 additional sections. In section 2, there is an outline of some important skeletonisation concepts and the algorithms used in the tests. The details of the proposed algorithm are described in section 3. Section 4 shows the results obtained in the tests carried out. Finally, in section 5, we conclude with observations and recommendations for future works.

2 Skeletonisation

In an image, skeletons are very useful for the recognition of elongated objects or patterns that have a certain shape, such as characters, polygons, chromosomal patterns, etc. Skeletons provide an extraction of topological and geometrical features of the object, so that when it is stored and processed, certain structural information about the original object is considered. Skeletonisation can be seen as a data compression process. The concept of skeleton was introduced by Blum in 1967 [6] in his analogy of middle axis detection with a grass fire. Since then, his definition has been used as a model for skeletonisation. A great number of techniques to obtain skeletons from discrete objects have been developed in the fields of Computer Vision and Pattern Recognition. Said techniques can be grouped into four different classes [7]: topological reduction [19] [28], distance transformation [9] [12] [20], curve evolution [16] [25] [27] and computational geometry [2] [22] [21] methods.

The skeletonisation technique based on topological reduction is frequently used to get the skeleton from a shape or object through thinning. Thinning is the reduction process of a digital image made up of certain number of pixels into a simplified version based on single-pixel-width line segments, so that the elimination of said point will not affect image connectivity and will respect the local end-point property in such a way that the topological properties of the object are preserved. In other words, after the pixels have been removed, the pattern must be recognized. The thinned version of a shape is called a skeleton. Fig. 1 shows different types of matrices (rectangular, hexagonal and triangular) used for pixel analysis [11] [10]. Likewise, sequential [15] [23] [18] and parallel [30] [14] [8] implementations have been published. In sequential algorithms, the pixels are eliminated in every iteration in a fixed sequence. The exclusion of a point p in iteration n depends on all operations executed until then. On the other hand, in parallel algorithms, the elimination of iteration n depends solely on the pixels of iteration $n-1$. Therefore, all pixels can be analyzed independently in parallel to each iteration.

Generally a rectangular matrix topology is used to generate a topological reduction. On the one hand, topological reduction can guarantee connected skeletons. However, most reduction algorithms can not always guarantee perfectly thinned shapes, since there will be cases where an array of pixels cannot be

Fig. 1. Matrices used for pixel analysis. (a) Rectangular matrix. (b) Hexagonal matrix. (c) Triangular matrix.

more thoroughly eroded. Moreover, these methods are seriously affected when objects undergo rotation. Nonetheless, skeletons produced with most techniques are sensitive to noise or to a variation of boundary, which often generates redundant skeleton branches that can alter the topology of the skeleton. To offset this problem, many skeletonisation algorithms include pruning methods, which appear as application-dependent [3]. Krinidis and Chatzis [17] have recently worked in an algorithm, without the use of any pruning methods, which does not generate spurious branches.

Aslan et al. [1] presented a different skeletal representation which deals with the problem of shape recognition with local deformations (see Fig. 2). Said algorithm relies on the stable features of the shape, instead of on the secondary inaccurately measured details. Therefore, the generated skeleton works with disconnected branches. The new representation does not suffer the common instability of the traditional connected skeletons, thus producing descriptions that are sensitive to any combination of changes in scale, position, orientation and articulation, as well as invariant ones. This way, the skeletons produced are similar, even when image boundaries undergo deformation or when there is a change in scale or rotation. From these data (location of disconnection and its length or branch), we can define primitives that can attain shape recognition through trees or skeletal graphs, where shape dissimilarity is calculated through distance correction.

The most important challenge for skeletal similarity is probably the fact that, on the one hand, the topological structure of trees or skeletal graphs of similar objects can be completely different (as a consequence of not taking into account the context). On the other hand, skeletal graphs of different objects can have

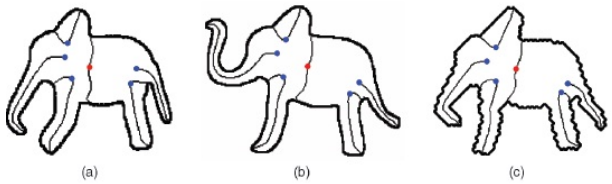

Fig. 2. Disconnected skeletons for the elephant in images with different rotation and borders. Notice that the branch and the location of the disconnection (indicated by a point) are similar [1].

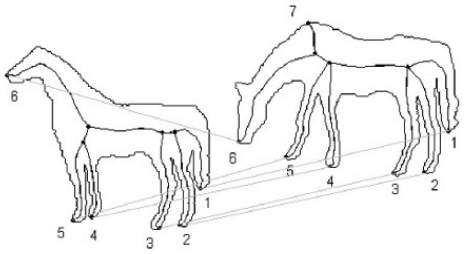

Fig. 3. The corresponding end nodes between the two skeleton graphs are linked with lines [4]

similar topology. To tackle these problems, Baseski, Erdem and Tari [5] exhibit an approach in which the contextual effects are considered relevant information for the calculation of skeletal similarity without being directly related to the geometric properties of the compared form. Likewise, Bai and Latecki's [4] main idea is to match skeletal graphs comparing geodesic paths between the skeleton's end-points without thinking about the topological structure of the graphic (see Fig. 3c).

Rizvandi, Pizurica and Philip [24], in their method for the detection of worms, decompose the skeleton into branches through the elimination of junction pixels (pixels with more than two neighbors), then calculate the angles for all branches and compare the angles of neighboring branches. Neighboring branches with an angle difference of less than a threshold are connected. Thus, a series of points (final, connecting and junction) and branches (final and connective) are defined.

3 Key Point Identification Algorithm

Our algorithm simplifies the skeletons previously obtained through any reduction technique in which skeletal thickness has a maximum width of one pixel. For instance, in Fig. 4a we show one input sample of the "LEMS 99 Silhouette Database", and in Fig. 4b you have the output of the Hilditch skeletonisation algorithm. Our proposal is part of the post-processing phase, which is applicable to skeletons. Notice that when an image from a two-dimensional object is thinned, the resulting skeleton has an irregular shape, based on arcs and curves. We expect to take that image and simplify it into points, obtaining angles, line segment sizes and proportions. For verification purposes, once the skeleton is simplified, we trace said points into perfect "straight" line segments by changing the coarse line segments of the original skeleton. We decompose an image into line segments. To do this, we define:

- End-point: A pixel (point) of the skeleton with only one neighbor.
- Intersecting-point: A pixel (point) in which two or more line segments cross or intersect.

These are key points within the structure of the skeleton. Our tests reveal a decrease in the amount of information necessary to represent a skeletonized

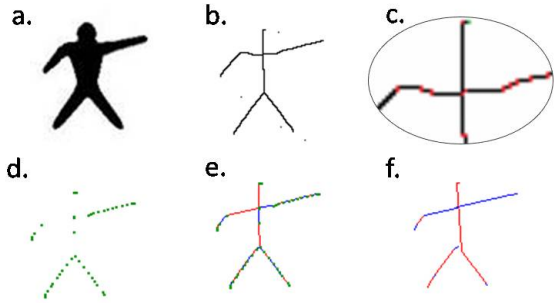

Fig. 4. (a) A sample from the "LEMS 99 Silhouette Database". (b) Result of the Hilditch skeletonisation algorithm. (c) Result of step "Finding out straight line segments". (d) Result of step "Finding out key points". (e) Result of step "Joining key points". (f) Result of step "Joining resulting line segments".

image, thus allowing us to center image analysis on said points. It is necessary to start from a previously obtained skeleton before running our processing algorithms. Any already well-known thinning algorithm can be used. Starting from that thinned image, we reduce the skeleton to points (expressed as pixel coordinates), and later we proceed to reconstruct the skeleton. Our algorithm consists of 4 significant steps, namely, finding out straight line segments, finding out key points, joining key points and joining resulting line segments.

3.1 Finding Out Straight Line Segments

The image is decomposed into straight line segments l, made up of consecutive pixels aligned in the same direction, containing a minimum of 2 pixels to represent a line segment. The slope for each line segment is yielded by function $m(l)$. For this, we have defined 4 directions: horizontal, vertical, and diagonal line segments slanting to the right and to the left. Then, two consecutive points $(x1, y1)$ and $(x2, y2)$ are aligned if:

- Horizontal (0°): $x2 = (x1 + 1)$ and $y2 = y1$
- Vertical (90°): $x2 = x1$ and $y2 = (y1 + 1)$
- Diagonal slanting to the right (45°): $x2 = (x1 + 1)$ and $y2 = (y1 + 1)$
- Diagonal slanting to the left (135°): $x2 = (x1 - 1)$ and $y2 = (y1 + 1)$

Through an iterative process, we search for continuous pixels all in the same direction, obtaining this way line segments longer or equal to 2 pixels. All the line segments found are stored with their starting (x_s, y_s) and final (x_f, y_f) coordinates in a set as shown in equation 1. In Fig. 4c, we show the output of this first step of the algorithm. Notice that all the disconnected points have been eliminated.

$$L = \{l^1[p^1(x_s, y_s), p^1(x_f, y_f)], ..., l^{max_l}[p^{max_l}(x_s, y_s), p^{max_l}(x_f, y_f)]\} \quad (1)$$

3.2 Finding Out Key Points

In this step, two types of key points, $p_k(x, y)$, are detected: end-points, $p_e(x, y)$, and intersecting-points, $p_\cap(x, y)$. End-points are those that have only one neighboring skeleton pixel. That is to say, the points where the continuity of the skeleton ends. This way $p_e(x, y)$ is an end-point if the 8-connectivity of pixel (x, y) is equal to 1 neighbor. All end-points are stored in set $P_e = \{p_e^1(x, y), ..., p_e^{max_e}(x, y)\}$. Also, the line segments related to all end-points are stored as sets $R_e(p_e^\kappa) = \{l^i, ..., l^j\}$.

The other group of key points is that of the intersections. This group represents the points where two or more line segments with different slopes cross. Let l' be a line segment yielded by points $p'_1(x'_1, y'_1), p'_2(x'_2, y'_2)$, and l'' a line segment yielded by points $p''_1(x''_1, y''_1), p''_2(x''_2, y''_2)$. $p_\cap(x, y)$ would be an intersection point for l' and l'' if,

$$m(l') \neq m(l'') \tag{2}$$

obtaining x and y coordinates as:

$$x = \frac{(y''_1 - y'_1) + (m(l') \times x'_1) - (m(l'') \times x''_1)}{m(l') - m(l'')} \tag{3}$$

$$y = m(l') \times (x - x''_1) + y'_1 \tag{4}$$

if, and only if, point $p_\cap(x, y)$ coincides with:

$$\begin{aligned} x'_1 \leq x \leq x'_2 \\ y'_1 \leq y \leq y'_2 \\ x''_1 \leq x \leq x''_2 \\ y''_1 \leq y \leq y''_2 \end{aligned} \tag{5}$$

All the intersecting are stored in another set $P_\cap = \{p_\cap^1(x, y), ..., p_\cap^{max_\cap}(x, y)\}$. Also, the line segments related to all intersecting-points are stored as sets $R_\cap(p_\cap^\mu) = \{l^k, ..., l^n\}$. Fig. 4d shows the 40 key points detected for our running example at this step.

3.3 Joining Key Points

In this step, we get the most significant result when the two key points are joined (end-points and intersecting-points), generating new line segments (l^u) that allow us to represent the original structure with a lot less information. To do this, a line segment is created between each pair of points if, and only if, there is a path between them and there are no key points between them. The process to join two key points is:

$$
\begin{aligned}
&L' = \emptyset \\
&\forall p_k(x,y) \in P_k \\
&\quad \forall l^i \in R_k(p_k(x,y)) \\
&\quad\quad \forall l^j \in L, l^j \neq l^i \\
&\quad\quad if \;\; \exists l^j | (p^i(x_f,y_f) \in l^i) \; and \; (p^j(x_s,y_s) \in l^j) \; are \; 8-connected \; then \\
&\quad\quad\quad if \;\; l^j(x_s,y_s) \in R_k \; then \\
&\quad\quad\quad\quad L' = L' \cup \{l^u[p^i(x_s,y_s), p^j(x_f,y_f)]\} \\
&\quad\quad\quad else \\
&\quad\quad\quad\quad l^i = \{l^u[p^i(x_s,y_s), p^j(x_f,y_f)]\} \\
&\quad\quad else \\
&\quad\quad\quad L' = L' \; \cup l^i
\end{aligned}
$$

To start, we define a set (L') to store the new line segments. Then, each line segments associated to some key point is compared to each line segment in the figure (L). That comparison allows to determinate if the line segments are 8-connected. In this case, a new line segment is created from their union. Finally, the line segment is stored in L'. Fig. 4e shows the way the key points have been joined trough the algorithm at this step.

3.4 Joining Resulting Line Segments

This is a polishing step where we detect line segments that can join and become one, from among the line segments generated in the previous step, and similarly, two line segments will join if both are 8-connected and have the same slope, resulting one line segment made up of the two most distant points, from among the four points that characterize both line segments. Fig. 4f shows the 14 resulting line segments.

4 Data and Results

The shapes from the 99−silhoutte database [24] were used by Goh [13], Sebastian, Klein, and Kimia [26], among others, in their experiments. In our case, the simplification algorithm was tested in 10 images (Fig. 5a), in which the skeletonisation algorithms previously described (Fig. 5b and e) were applied and from which the resulting images were obtained. Finally, the process results in a list of coordinates that make up the resulting line segments. Therefore, we can obtain line segments' sizes, proportions between line segments and their respective angles of inclination, and specially a skeleton with the morphology of the original structure but with less information (Fig. 5c and f). In the same way, Fig. 5 in columns (c) and (g) shows the reduction rate obtained by our technique. The reduction is calculated by dividing the number of lines needed to represent the skeleton after our process by the number of the original lines.

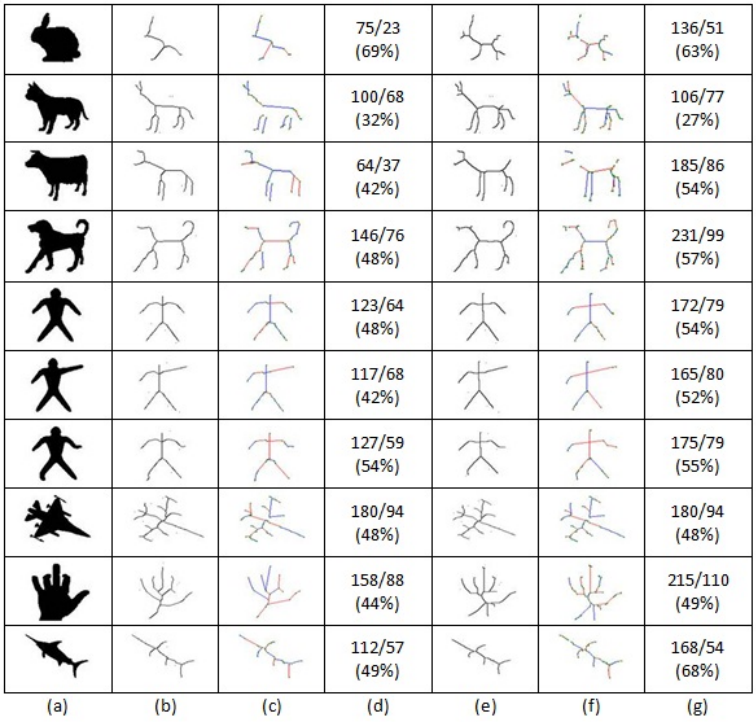

Fig. 5. (a) Shapes used in the experiments. (b) Results of the Zhang-Suen skeletonisation. (c) Final results of our algorithms on Zhang-Suen skeletonisation. (d) Reduction rate compared to the Zhang-Suen algorithm. (e) Results of the Hilditch skeletonisation. (f) Final results of our algorithms on Hilditch skeletonisation. (g) Reduction rate compared to the Hilditch algorithm.

5 Conclusions

The use of skeletons and their subsequent polishing allows for data compression, reducing the need for storing, as well as improving the quality of the information stored, since it dismisses irrelevant data generated by common skeletonisation algorithms, such as isolated pixels. Many methods to skeletonized images have been developed. Goh [13] recently proposed an image comparison method using skeletons. This paper proposes a method that allows us to simplify, not only data from a skeleton, but also its subsequent analysis, resulting in a series of related coordinates, which represent the skeletonized image in a reliable way without the computational cost of analyzing a complete image.

We have shown that the skeleton represented in the simplified image can be reconstructed, with a high degree of accuracy, based on the points (coordinates) generated in the simplification process. The problems that have come up are previous to algorithm skeletonisation, specifically to the generation of line segments that are more than one pixel wide. In future works, the effectiveness of

the method will be considered when the skeletonized image undergoes alterations related to rotation, position and scale. The refinement method will also extend to three-dimensional images, keeping the method as simple as possible. Lastly, remember that the algorithms proposed are a first step towards human activity recognition through the analysis of human poses represented by their skeletons.

Acknowledgements

This work was partially supported by Spanish Ministerio de Ciencia e Innovación under projects TIN2007-67586-C02-02 and TIN2010-20845-C03-01, and by Junta de Comunidades de Castilla-La Mancha under projects PII2I09-0069-0994, PII2I09-0071-3947 and PEII09-0054-9581. The authors are grateful for the public use of the *LEMS 99 Silhouette Database*.

References

1. Aslan, C., Erdem, A., Erdem, E., Tari, S.: Disconnected skeleton: Shape at its absolute scale. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(12), 2188–2203 (2008)
2. Aurenhammer, F.: Voronoi diagrams: A survey of a fundamental geometric data structure. ACM Computing Surveys 23(3), 345–405 (1991)
3. Bai, X., Latecki, L.J., Liu, W.Y.: Skeleton pruning by contour partitioning with discrete curve evolution. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(3), 449–462 (2007)
4. Bai, X., Latecki, L.J.: Path similarity skeleton graph matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(7), 1282–1292 (2008)
5. Baseski, E., Erdem, A., Tari, S.: Dissimilarity between two skeletal trees in a context. Pattern Recognition 42(3), 370–385 (2008)
6. Blum, H.: A transformation for extracting new descriptors of shape. In: Models for the Perception of Speech and Visual Form, pp. 153–171. MIT Press, Cambridge (1967)
7. Bouix, S., Siddiqi, K.: Optics, mechanics, and Hamilton-Jacobi skeletons. Advances in Imaging and Electron Physics 135, 1–39 (2005)
8. Chin, R.T., Wan, H.K., Stover, D.L.: A one-pass thinning algorithm and its parallel implementation. Computer Vision, Graphics, and Image Processing 40(1), 30–40 (1987)
9. Danielsson, P.: Euclidean distance mapping. Computer Vision, Graphics, and Image Processing 14, 227–248 (1980)
10. Davies, E.R., Plummer, A.P.N.: Thinning algorithms: A critique and a new methodology. Pattern Recognition 14, 53–63 (1981)
11. Deutsch, E.S.: Thinning algorithms on rectangular, hexagonal, and triangular arrays. Communications of the ACM 15(9), 827–837 (1972)
12. Ge, Y., Fitzpatrick, J.M.: On the generation of skeletons from discrete euclidean distance maps. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(11), 1055–1066 (1996)
13. Goh, W.B.: Strategies for shape matching using skeletons. Computer Vision and Image Undertanding 110(3), 326–345 (2008)

14. Hall, R.W.: Fast parallel thinning algorithms: Parallel speed and connectivity preservation. Communications of the ACM 32(1), 124–131 (1989)
15. Hilditch, C.: Linear skeletons from square cupboards. Machine Intelligence 4, 403–420 (1969)
16. Kimia, B.B., Tannenbaum, A.R., Zucker, S.W.: Shapes, shocks, and deformations I: The components of two-dimensional shape and the reaction-diffusion space. International Journal of Computer Vision 15(3), 189–224 (1995)
17. Krinidis, S., Chatzis, V.: A skeleton family generator via physics-based deformable models. IEEE Transactions on Image Processing 18(1), 1–11 (2008)
18. Kwok, P.C.K.: A thinning algorithm by contour generation. Communications of the ACM 31(11), 1314–1324 (1988)
19. Lam, L., Lee, S.W., Suen, C.Y.: Thinning methodologies: A comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(9), 869–885 (1992)
20. Leymarie, F., Levine, M.D.: Simulating the grassfire transform using an active contour model. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(1), 56–75 (1992)
21. Ogniewicz, R.L., Ilg, M.: Voronoi skeletons: Theory and applications. In: Proc. Conference on Computer Vision and Pattern Recognition, pp. 63–69 (1992)
22. Ogniewicz, R.L., Kübler, O.: Hierarchic voronoi skeletons. Pattern Recognition 28(3), 343–359 (1995)
23. Pavlidis, T.: A thinning algorithm for discrete binary images. Computer Vision, Graphics, and Image Processing 13, 142–157 (1980)
24. Rizvandi, N.B., Pizurica, A., Philips, W.: Automatic individual detection and separation of multiple overlapped nematode worms using skeleton analysis. In: Campilho, A., Kamel, M.S. (eds.) ICIAR 2008. LNCS, vol. 5112, pp. 817–826. Springer, Heidelberg (2008)
25. Scott, G.L., Turner, S.C., Zisserman, A.: Using a mixed wave/diffusion process to elicit the symmetry set. Image and Vision Computing 7(1), 63–70 (1989)
26. Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of shapes by editing shock graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(5), 550–571 (2004)
27. Tari, S., Shah, J., Pien, H.: Extraction of shape skeletons from gray-scale images. Computer Vision and Image Understanding 66(2), 133–146 (1997)
28. Xie, W., Thompson, R.P., Perucchio, R.: A topology-preserving parallel 3D thinning algorithm for extracting the curve skeleton. Pattern Recognition 36(7), 1529–1544 (2003)
29. Xu, W., Wang, C.X.: A fast thinning algorithm implemented on a sequential computer. IEEE Systems, Man, and Cybernetics 17(5), 847–851 (1987)
30. Zhang, T.Y., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. Communications of the ACM 27(3), 236–239 (1984)