# Semi-Automatic Delineation of Regions in Floor Plans

Kathy Ryall
*Harvard University*

Joe Marks
*Digital Equipment Corporation*

Murray Mazer
*Digital Equipment Corporation*

Stuart Shieber
*Harvard University*

November 29, 1994

**Contact:**
Stuart M. Shieber
Division of Applied Sciences
Harvard University
33 Oxford Street— 14
Cambridge, MA 02138
phone: (617) 495-2344
fax: (617) 496-1066
email: `shieber@das.harvard.edu`

# Semi-Automatic Delineation of Regions in Floor Plans

**Abstract**

The ability to recognize regions in a bitmap image has applications in various areas, from document recognition of scanned building floor plans to processing of scanned forms. We consider a technique that makes use of a kind of proximity metric for delineating partially or fully bounded regions of a scanned bitmap that depicts a building floor plan. A proximity field is defined over the bitmap, which is used both to identify the centers of subjective regions in the image and assign pixels to regions by proximity. The region boundaries generated by the method tend to match well the subjective boundaries of regions in the image. We discuss incorporation of the technique in a semi-automated interactive system for region identification in floor plans. In contrast to area-filling techniques for delineating areal regions of images, our approach works robustly for partially bounded regions. Furthermore, the frailties of the method that do remain, unlike those of alternative techniques, are well-moderated by simple human intervention.

**Keywords:** 2-D graphics, drawing software, region finding, deformable templates, building geography.

# 1 Introduction

Location-aware technology and ubiquitous computing are major new trends in applications computing (Weiser, 1991; Wellner, Mackay, and Gold, 1993). One premise of this work is that the ability to track the movement of people and artifacts (e.g., computers, machinery, vehicles, etc.) will make possible new computer applications for better managing the home and workplace. Most current research in this area is focused on developing the enabling hardware (e.g., the Digital-Olivetti Active Badge system (Want and Hopper, 1992; Want et al., 1992; Fishman and Mazer, 1992)). Less attention has been paid to software. Because location-aware computing depends upon the specific environmental or physical context in which it is employed, it requires software to support the input, representation, processing, and visualization of such physical settings. In this paper we focus on the input problem: our broad goal is to develop software techniques to facilitate the input of information about building geographies that is needed for location-aware computing.

In general, geographic information for buildings is available only from hard-copy floor plans.[1] (A sample bitmap scanned from a floor plan is shown in Figure 3a.) Thus the data is not in symbolic form; it must be interpreted visually, a task that is easy for people but not for computers. The information required for location-aware computing is *region data*: the topology and geometry of "people" regions (e.g., offices, lobbies, or closets) and "computer" regions (e.g., the areas scanned by

[1]Another potential source of geographic data for buildings is architectural CAD systems. However, this data source is less attractive than it might seem at first blush because data are available for recently constructed or renovated buildings only, and various systems use different data formats and representations.

sensing devices). To extract this data from a scanned floor plan requires annotating the floor plan to delineate the relevant regions. The problem of region delineation is the one we address in this paper.

## 1.1 Existing Approaches

Several software tools exist today that can facilitate the delineation of areal regions on a floor plan. They all utilize one or more of the following four techniques:

1. *Freehand drawing:* The user redraws the floor plan from scratch with a mouse or digitizing tablet, delineating regions as part of the drawing process.

2. *Tracing:* The user traces out regions on the floor plan by hand, using a mouse or digitizing tablet.

3. *Area filling:* The user selects points on the floor plan image, and the computer fills in the fully enclosed areas surrounding the points. (The method is often called "flood filling".)

4. *Template fitting:* The user positions and sizes predefined templates (usually rectangles) to cover the desired regions approximately.

All of these techniques have drawbacks. Freehand drawing is time-consuming, tedious, and error prone. Tracing is equally inefficient, though more accurate. Area filling would appear at first sight to be fast and accurate, but it only works for fully enclosed regions: if the user wants to specify a region that is partially bounded — the boundaries of the region might have been corrupted during the scanning process

(a frequent occurrence), or the region might be bounded in part by subjective contours not depicted in the image — the filling technique will not work because the filling process will "leak out" to the rest of the image through missing boundaries or holes in existing boundaries. Finally, template fitting, though fast, is inherently inaccurate.

## 1.2 Comparing Approaches

Unlike the three manual methods, the area-filling approach has the attractive property of being semi-automatic: The user chooses the point from which to start the filling process, which is then performed automatically. However, as we have seen, the technique is brittle, as it relies on the unrealistic assumption that *subjective boundaries* in an image (the boundaries of image areas that are subjectively perceived as forming distinct regions) coincide with actual closed pixel contours in the image.

Of course, the task of region delineation, like all nontrivial document recognition tasks, can depend in the end on arbitrary semantic information about the material depicted in the bitmap that no purely syntactic method can be sensitive to. For example, the distinction between a foyer to a room and a bay in it may be unmarked in a floor plan. If the former is considered a separate region and the latter part of the same region, only someone familiar with this semantic distinction would be able to delineate the regions correctly. Thus, the inability of area-filling to work perfectly is in a sense inevitable. Because no fully automatic method is to be expected, we believe that it is crucial to think of the task of region delineation as being

solved only semi-automatically. Thus, any system for region delineation needs to be evaluated not only for how well it works in an absolute sense, but also, how well any remaining problems (for there will be some) can be easily handled by simple human interventions. The articulation between the automatic technique and the human intervention in a semi-automated system is crucial to the appropriateness of a region delineation technique.

Viewed from this perspective, the area-filling technique is seen to be especially lacking, for when it fails, it fails in a way that provides no help to the user. Although some automatic preprocessing of the image may help correct some area-filling errors, any remaining errors will require either careful manual cleanup of the image. Alternatively, intervention in the form of postprocessing of a region incorrectly delineated by filling is hardly easier than specifying the region by one of the fully manual methods.

We propose an alternative method that utilizes a similar division of labor between user and computer but relaxes the assumption that subjective boundaries and pixel contours coincide. Pixels are assigned to regions by making use of a *proximity field*, the local minima of which define the centers of regions and the valleys of which represent the regions themselves.

Relative to area filling, the proximity field method has the advantage that region delineation is robust with respect to corrupted or missing boundaries in the image. Furthermore, where the method fails, simple human interventions requiring only gross information can be used to correct the result. Finally, although full discussion is beyond the scope of this paper, the technique can take other application-specific

region characteristics into account.

In the remainder of the paper we describe the technique of proximity fields for region delineation (Section 2), the use of simple human intervention to correct the frailties of the system (Section 3), and a prototype system that we have built to test our algorithm (Section 3.1) including some illustrative examples (Section 3.2).

# 2 The Proximity Field Technique

The novel contribution of our approach is the method by which the subjective boundaries of regions are defined. As discussed above, the standard area-filling method has the problem that subjective boundaries that are not marked with explicit lines in the image are overrun so that neighboring regions are invaded. Our approach attempts to better characterize the notion of a subjective region by encapsulating the notion "subjective region boundary" in a single function and by viewing the region-identification problem as the problem of optimizing this function.

The approach can be motivated by reconstructing the problem with the area-filling method. Suppose we are given a drawing of the two-room building given in Figure 1a. Note the door between the rooms. Because of this door, a filling algorithm started from any point would find a single region (as in Figure 1b).

Intuitively, the reason that a given pixel, say the one labeled $r$ in Figure 1f, is taken to be associated with the right room rather than the left (where the filling was started) is that it is *closer* to the *center* of the right room than the left room. In order to use this intuition to actually delineate regions, we must find a way of

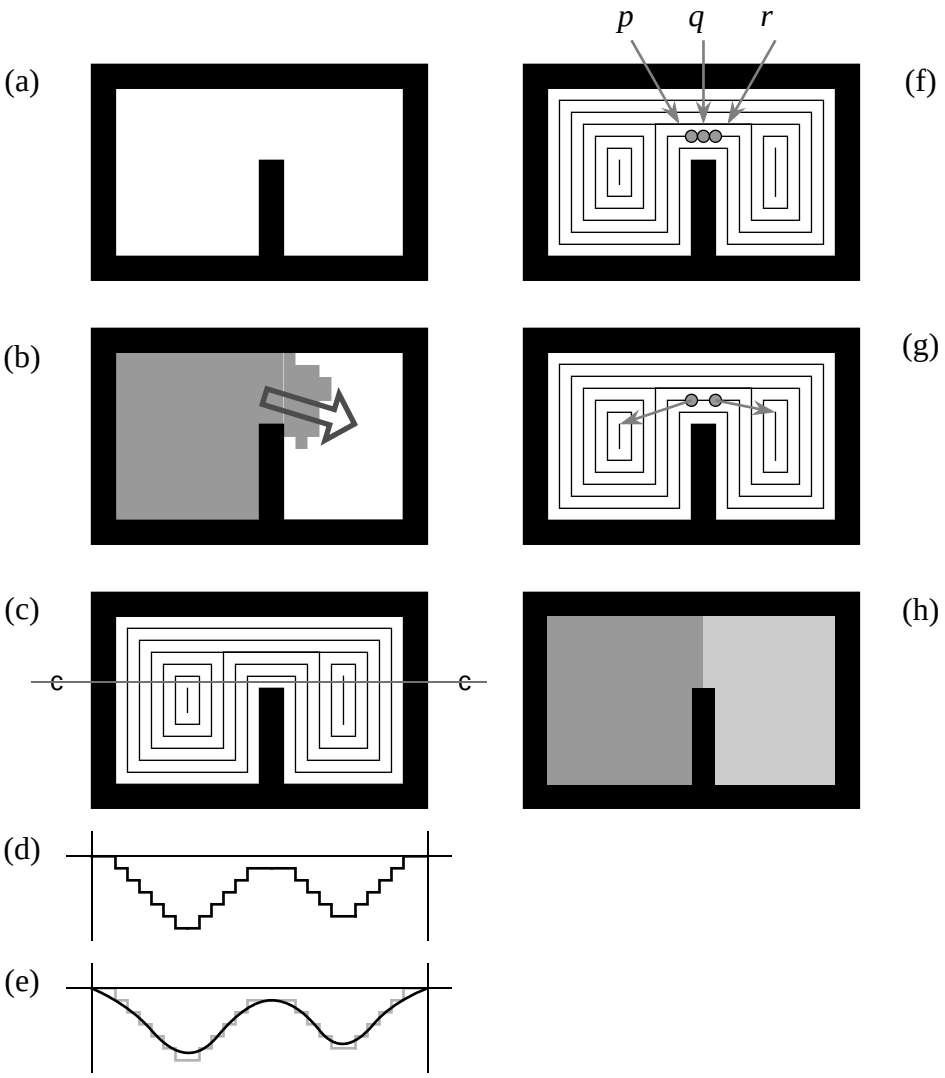



Figure 1: (a) A simple two-room drawing. (b) A filling algorithm started in the left room traverses the subjective boundary of the door. (c) A "topographic map" of the proximity field for the floorplan. (d-e) Unsmoothed and smoothed cross-sections of the proximity field along a horizontal line through the doorway. (f) Points to the left of the doorway (as point $p$) are nearest to the local minimum in the left region, whereas points to the right (as $r$) are closest to the local minimum in the right region. Points right in the middle (as $q$) may be considered part of either room; the algorithm presented in Section 2.1 would associate such a point with the deeper minimum, hence the larger left room. (g) The notion of closeness can be motivated by imagining a ball located on the surface, rolling downhill to the nearest local minimum. (h) Regions defined by the proximity field technique shown as a two-coloring of the floorplan.

characterizing these two notions of 'closer' and 'center'.

We do so with a proximity field. Imagine a surface defined so that the height of the surface at each black pixel in the image is zero and at each interior pixel the surface is as many units below zero as the pixel's distance to the nearest black pixel, so that the surface forms a series of valleys with the black pixels as ridges separating them. The surface just defined is the *proximity field*; a topographic map of a surface is given in Figure 1c and a cross-section is shown in Figure 1d. Local minima in this field provide a rough characterization of the notion *center of a region.*

We want to assign each pixel in the image to one of the field minima, in particular, the closest one. The appropriate notion of closeness is not mere geometric distance (as defined by, say, a "Manhattan distance" metric). Instead, the surface itself provides a definition of closeness. Objects on such a surface tend to move downhill so as to locally minimize their potential energy. The minimum reached from a given pixel by such a local minimization process is an appropriate notion of "closest center". We can imagine a ball placed at the given pixel and released; the local minimum that it settles in is the center of the room the pixel belongs to. Since, as depicted in Figure 1g, a ball at point $p$ would roll to the upper minimum, whereas one at point $r$ would roll to the lower minimum, the two points are taken to lie within the upper and lower region respectively.

This rolling ball analogy fails, unfortunately, when the pixel in question is on a plateau of the surface between two minima, say at point $q$ in Figure 1f, in which case both minima are reachable by descent from the given pixel. Conceptually, this aberration can be eliminated by using a smoothed version of the surface as depicted

in Figure 1e. In practice, the allocation of pixels to regions that would be engendered by using the smoothed surface can be calculated directly by the technique described in Section 2.1, without actually calculating the smoothed field.

This discussion provides evidence that an approach based on energy minimization in the proximity field might be appropriate for characterizing subjective boundaries better than the simple filling technique. Our method is based on just this metaphor of energy minimization. We describe the method in more detail in the next section.

## 2.1 Computing Minima of the Proximity Field

A region, under the model we are describing, is characterized as the set of pixels for which a given minimum in the proximity field is closest. Thus, specifying a region follows from specifying the corresponding field minimum. This is easily done by local search. Given a set of pixels all of equal value, which we will call the *working set*, a new set is generated in the following manner. The neighboring pixels with the lowest field value are found. If the pixels in this neighboring set are higher than those in the current set, the current set constitutes (part of) a local minimum of the surface, and the search is done. If the neighboring set pixels are the same height, the working set is augmented with the pixels in the neighboring set, and the process is iterated. If the neighboring set pixels are lower, the neighboring set becomes the current set and again the process is iterated. Eventually, this iterative search terminates with a set of equal-value pixels that comprise a minimum. A graphical depiction of the iterative process is presented in Figure 2.

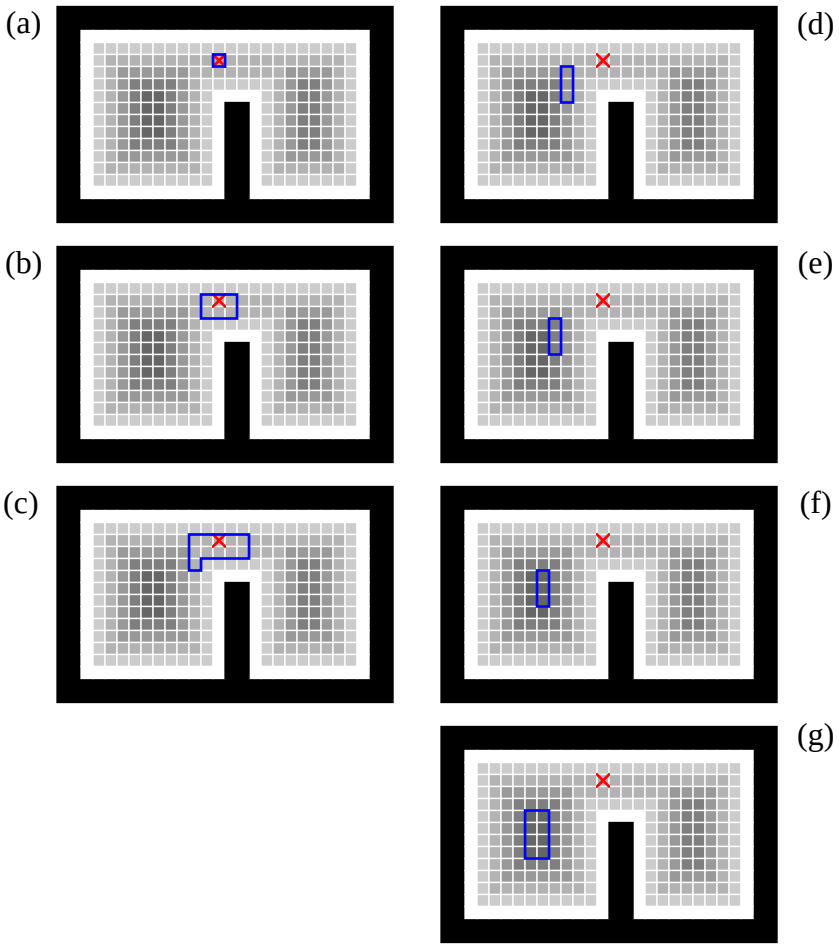



Figure 2: Stages in the computation of the nearest local minimum to point $p$ of Figure 1f. The proximity field is given as a false grayscale coloring of the interior pixels, with darker colors being lower. The starting pixel is marked with an $\times$ in all drawings, and the pixels of the working set is bounded with a polygon. (a) The working set is initialized to the starting pixel. (b-c) The lowest-valued neighboring pixels are of the same value, so are added to the working set. (d-f) The lowest-valued neighboring pixels are of lower value, so they become the new working set. (f) Pixels within the local minimum has been found. (g) The working set grows to encompass the full local minimum, and the algorithm halts.

Under certain conditions, for instance, when started at point $q$ in Figure 1f, the working set may become temporarily noncontiguous. In the area within the door, for instance, there is a plateau in the proximity field. The first few iterations of the algorithm above expand the working set to encompass larger areas of the plateau. Eventually, if the starting point is nearer one room or the other, the edge of the plateau nearer that side will be found, and the working set will move in that direction, eventually settling in the corresponding minimum. However, if the starting point is in the exact center of the plateau, both edges will be found on the same iteration, and the working set will comprise pixels from both sides of the plateau. The search will continue in this way until one of the discontiguous parts of the working set hits a minimum. If the other part is not yet at a minimum, the search will continue, using only the latter portion of the working set, eventually finding the lower of the two minima. In essence, the method as described places points midway between two rooms as part of the larger of the two rooms. (If both minima are reached at the same time, that is, both rooms are the same size, one or the other can be chosen arbitrarily.)

# 3 Incorporation in a Semi-Automated System

The basic proximity field technique works quite well for finding subjective contours of regions in bit maps. (Section 3.2 provides examples of the method as applied to actual scanned floorplans.) However, as argued in Section 1.2, the articulation of any such method with some human intervention ability is crucial. The essential

idea behind leveraging of human intervention is simply this: repairing of incorrect or inaccurate results of the automatic method should require only gross human interaction, rather than finely detailed work. The proximity field technique compares well with other techniques, such as area filling, not only in that it performs better *ab initio* at finding subjective contours, but also because it lends itself well to this leveraging the addition of simple human intervention.

The remaining frailties of the proximity field technique can be classified as follows:

**Missing subjective contours:** Indicators of a subjective contour may be wholly or partly absent from the input image. For instance, what appears to be a single large room in the floor plan may be thought of by the occupants of the room as two separate regions. Since no syntactic reflex of the distinction between the two rooms is found in the image, no method based purely on bitmap processing can be expected to observe this distinction.

**Multiple minima:** A subjective region may encompass several minima, that is, it is the union of the regions defined by the multiple minima.

Both of these problems are easily handled by only simple human interventions, given an appropriate user interface.

Indicators of missing subjective contours can be manually added with simple line-drawing tools. Because the method does not require closure of the subjective contours, simple "hints" as to the missing subjective contour are all that is typically required to repair the region delineation. (For example, in Figure 4b in the next

section, a short line segment is all that is required to coerce the proximity field method to find the correct subjective contours.)

It might be thought that the ability to add lines or other features to the image as a preprocessing step would vitiate the problems with area filling. It does not. In the case of area filling, quite fine-grained manual preprocessing of the image will tend to be required, because the preprocessing would need to be used not only to add missing large-scale subjective contours but also any missing bit of subjective contour down to the pixel level. The improved automatic performance of the proximity field technique means that only the grossest of missing subjective contours need be filled in. Because the proximity field technique is relatively insensitive to perfect abutting of lines and the like, the line drawing can be done quite rapidly, and on an as-needed basis. As the user of a system sees that a certain region is not being appropriately subdivided, the user merely needs to provide a hint to the system by quickly drawing a line segment where there is a missing contour.

Subjective regions encompassing multiple proximity field regions can similarly be handled without fine-grained editing of the image. A user can specify to the system that several regions should be combined to form a single subjective region merely by indicating the corresponding centers. Because the local search procedure described in Section 2.1 can find region centers starting from a large area around the minimum, the process of selecting the regions can be done manually with only gross human actions, such as mouse-clicking in the general vicinity.

Thus, the proximity field technique not only works better as an automated method for region delineation, but it fits well in a semi-automated system in that (i)

the better performance means that less repair of results need be carried out manually, and (ii) the types of human intervention needed to correct the behavior of the technique require only large-scale gross actions rather than fine-grained editing.

Another feature of the proximity field technique is that alternative definitions of the proximity field might be used to characterize regions of a quite different sort from boundaries perceived as physically contained regions. The simple field used here models regions unconstrained in all directions except by overt indicators of subjective contours. The sensing region of a motion sensor, by contrast, is constrained to subtend a certain angle outward from the site of the sensor. The regions defined by such a sensor can be defined by an alternative proximity field that is more constrained than the one used here but that still respects subjective regions. The generality of the proximity field approach to region delineation thus makes possible the modeling of many different kinds of regions. Similarly, an alternative proximity field definition might be used to find the subjective segmentation of a document image.

## 3.1 The Prototype System

We have implemented the proximity field technique for region delineation in a prototype system. The interface for the system includes a viewing area for the bitmap image and a control panel for various parameters. The system computes the proximity field over the bitmap and the corresponding regions. The former can be displayed to the user as a grayscale topography of the surface. The latter is displayed as a false coloring of the regions.

The system provides for human intervention in the form of user-added subjective contour "hints" and region grouping as described in Section 3. No pixel level post-editing of the regions is supported; the need for post-editing at this level of detail is rare and would be better viewed as an indicator of a flaw in the method. All human intervention is achieved through standard graphical drawing and selection techniques that can be used at several magnifications of the image.

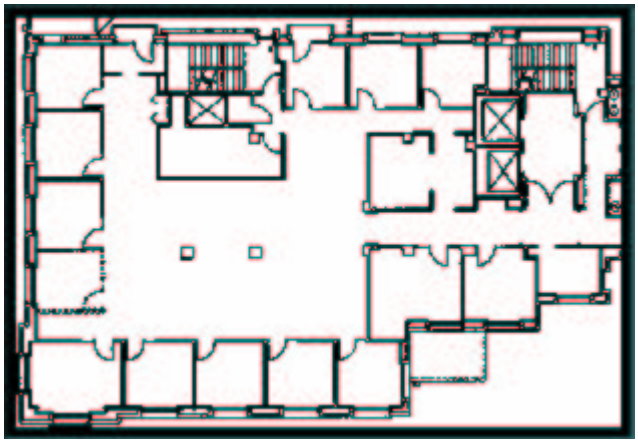
Once the user has accepted a delineation of the regions in the bitmap, the system will save a description of the regions. At this point, the user may annotate the regions with various properties. Typical characteristics of building regions include room numbers, names of occupants, telephone information, lighting characteristics, etc.

## 3.2 Examples

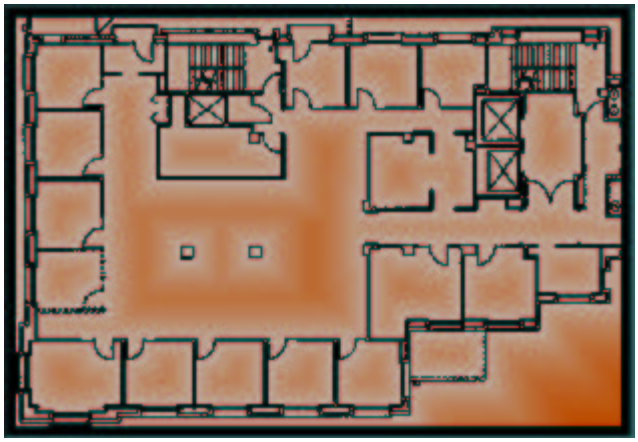
In order to provide a feel for the quality of region delineation achievable using the proximity field on actual scanned bitmaps, we apply it to a sample scanned floor plan, a map of the Digital Equipment Corporation Western Research Laboratory. The bitmap is shown as Figur 3a. Note that the boundaries of some of the rooms are discontinuous as a result of scanning errors; the lowest of the four rooms on the left wall is an example. Other rooms, for instance the room in the lower right corner, have open doors. Consequently, area-filling performs poorly on the image, as shown in Figure 4a.

The proximity field of the map is shown in Figure 3b. The value at each white pixel is shown through false-coloring, with darker tones connoting lower values.

(a)



(b)



(c)

Figure 3: Sample scanned bitmap. (a) The original bitmap. (b) The false-colored proximity field generated for the bitmap. (c) The regions delineated according to the proximity map in (b).

(a)

(b)

Figure 4: Regions delineated by two methods. (a) Regions generated by area-filling. (b) Regions generated by the proximity field method, with minor postediting. Note the introduction of the subjective contour "hint" at the upper right of the main lobby and the joining of regions in the room above it.

When regions are defined on the basis of this proximity field, the result, shown in Figure 3c, is much more accurate than the area-filling method. In particular, the rooms mentioned above with discontinuous borders are correctly delineated.

Nonetheless, several problems remain. Introduction of a single subjective contour "hint" and the joining of a few regions cleans up the iamge to form the final region delineation seen in Figure 4b.

# 4 Conclusion

Region delineation, like many document analysis problems, cannot be solved perfectly by purely syntactic methods. Consequently, any method for identifying regions in a bitmap must allow for human intervention at some point in order to correct errors detectable only with semantic information. The proximity field method, besides providing a better syntactic model of subjective region, also allows for simple interactive postprocessing; it is thus superior to both manual and automatic methods previously proposed.

# References

Ballard, Dana and Christopher M. Brown. 1982. *Computer Vision.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Cleary, John G. and Geoff Wyvill. 1988. Analysis of an algorithm for fast ray tracing using uniform space division. *Visual Computer*, 4(2):65–83.

Fishman, Neil and Murray S. Mazer. 1992. Experience in deploying an active badge system. In *Proceedings of the IEEE Globecom Workshop on Networking of Personal Communications Applications*, pages I–1–10, Orlando, Florida, December.

Kass, Michael, Andrew Witkin, and Demetri Terzopoulos. 1988. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331.

Want, Roy and Andy Hopper. 1992. Active badges and personal interactive computing objects. *IEEE Transactions on Consumer Electronics*, 38(1):10–20, February.

Want, Roy, Andy Hopper, Veronica Falcão, and Jonathan Gibbons. 1992. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, January.

Weiser, Mark. 1991. The computer for the 21st century. *Scientific American*, 265(3):94–104, September.

Wellner, Pierre, Wendy Mackay, and Rich Gold. 1993. Special issue on computer-augmented environments. *Communications of the Association for Computing Machinery*, 36(7), July.

Yuille, Alan L., Peter W. Hallinan, and David S. Cohen. 1992. Feature extractions from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111.