Extracting Geometric Information from Architectural Drawings

Brian W. Kernighan[1] and Christopher J. Van Wyk[2]

[1] Bell Laboratories, Murray Hill, NJ 07974, USA,
bwk@research.bell-labs.com
[2] Department of Mathematics and Computer Science, Drew University,
Madison, NJ 07940, USA, cvanwyk@drew.edu

Abstract. We describe some experiments on a geometric problem encountered while developing a tool for predicting the behavior of indoor wireless communications systems. The problem is to extract information about the walls of a building from a machine-readable description. These descriptions contain much irrelevant or redundant information; the task is to eliminate the former and to reduce the amount of the latter. A key geometric idea is to use plane sweep to find line segments in the drawing that belong to the same wall. A great gulf lies between this simple idea and its realization, however, and any solution must involve a tradeoff among factors like accuracy, speed, and the degree and nature of human intervention.

1 Introduction

For the past two years one of the authors has been part of a group developing the WISE (Wireless System Engineering) system for design and optimization of indoor wireless communications systems. Cellular and cordless phones, wireless networks, active badges, and radio-controlled shelf tags in supermarkets are among the many applications of this kind of system. WISE requires as input a description of the coordinates and composition of the walls of a building, and myriad parameters, including power, frequency, antenna types, and signal to noise ratios. A family of programs computes radio energy throughout the building, to predict the behavior of the system, to minimize system cost by varying the locations of transceivers, and to analyze coverage, sensitivity, and other properties. The overall system is described in [FGK+95].

Developing WISE required solutions to a variety of computational geometry problems. Techniques that reduce computational time from infeasible to practical are described in [F96]. This paper is about a smaller, subsidiary problem that illustrates some pragmatic issues that lie between the important ideas of computational geometry as viewed by the computer science community and the difficult real-world problems posed by potential users of those ideas.

The problem is to determine the coordinates and composition of the walls of the building from whatever representation is given. Although easy to state and to understand, this seems to be the most difficult problem faced by users of WISE. Recent buildings, say those built within the past decade or two, are often described in some computer-aided design system, usually AutoCAD ''DXF'' files [MvR94]. We attacked the problem of processing AutoCAD drawings mechanically to make them more suitable as building representations for WISE.

2 Decoding AutoCAD DXF Files

A DXF file contains a header section that defines parameters, a blocks section that defines drafting symbols and common constructs like door frames, and an entities section that contains all the items in the drawing. Each entity has a type, coordinates, color, and other parameters, and is associated with a ''layer'' of the diagram; the color and layer may indicate its role in the diagram or building.

Since DXF files are in ASCII, we can use standard Unix tools to manipulate them. Thus, a 150-line AWK [AKW88] program parses DXF files and outputs an ASCII representation of line segments: each segment is represented by the (x, y) coordinates of its endpoints, an assumed z-value, and any information about composition that can be guessed from the layer and color fields. This program also does any necessary coordinate transformations, like clipping, inverting y, and converting feet to meters. Other AWK programs process these line segments; the Unix `sort` command puts them into various useful orders; and Tcl/Tk [O94] lets us quickly write programs to view the data interactively before, during, and after processing.

3 Removing Extraneous Features

Figure 1 shows the ground floor of AT&T's Learning Center, a 3-story conference center and hotel in Basking Ridge, NJ. Although some information (like wiring and pipes) had already been filtered out before we got the data, it is apparent from what remains that DXF files are meant to produce drawings for people. They contain features like closely spaced, short line segments that represent stairs; arcs or lines that indicate the sweep of doors; cross-hatched elevator shafts and building columns; dotted and dashed grid lines; line drawings of plumbing fixtures; and textual annotations. For instance, the lines at the top of Figure 1 underline text that titles the drawing, and the ''arrowhead'' marks the nominal North.

Thus, part of the processing is to weed out such extraneous features. After some experiments, we arrived at the following heuristics:

- Discard all entities except LINEs and POLYLINEs.

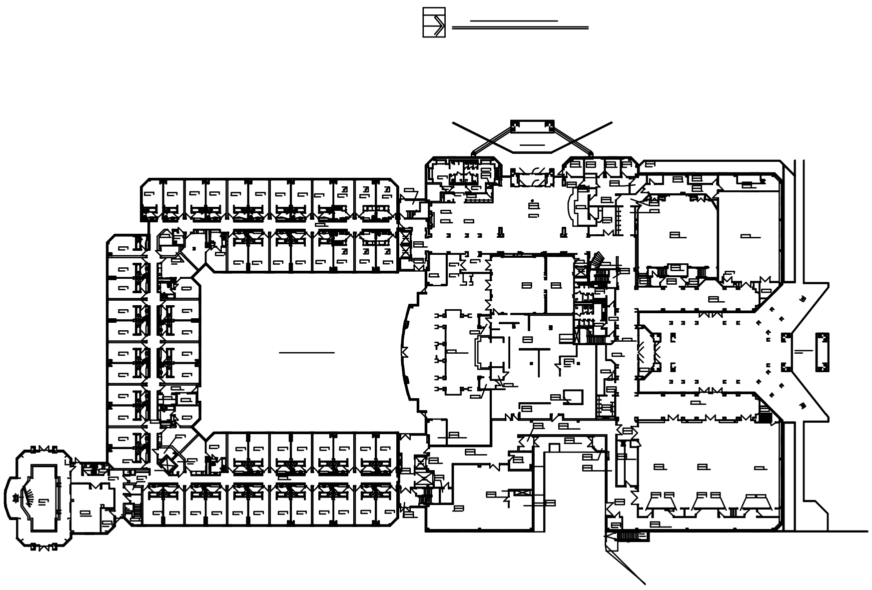


Fig. 1. AutoCAD drawing of AT&T Learning Center ground floor (~ 155m × 80 m)

- Eliminate all line segments that are too short to matter to the radio propagation model.
- Eliminate line segments that serve only to connect annotation text to a feature of the drawing; these often follow a TEXT entity directly and share an endpoint with it.
- Eliminate any line segment that is not drawn in CONTINUOUS mode; it is likely to be a grid line.

The first two heuristics are fairly general and likely to apply to many buildings. The second two, however, apply only to some of the buildings we have analyzed for WISE users, and might well need to be modified or augmented for other representations. Each new building — indeed, different floors of the same building — can add to a growing list of heuristics that have proven useful for removing extraneous features.

One lesson we learned from the exploration and experiments that led to these heuristics is that a DXF file is in no way a complete or accurate representation of a building. It is more realistic to regard the diagram as a comment: at best it bears some relation to some version of the building at some time, but it should not be confused with the real thing. (*Proof.* In Figure 2, only half the hotel rooms have toilets.)

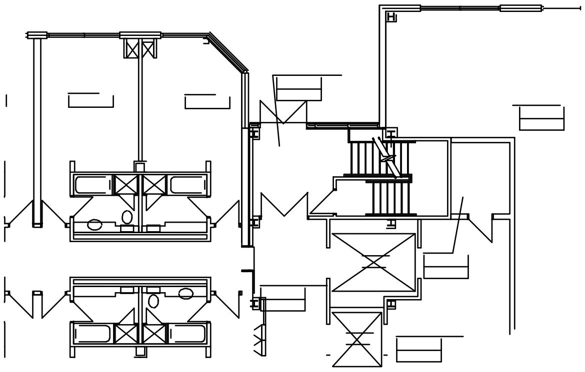


Fig. 2. Detail from Figure 1

4 Coalescing Nearby Walls

The close-up view in Figure 2 shows that the DXF file represents each wall as a collection of line segments that define the wall's perimeter in the plan view: each ''logical wall'' has at least two line segments, one for the room on each side of the wall. WISE, however, expects the building representation to contain ''logical walls,'' since it models each such wall as a plane surface with zero thickness but specified dielectric properties appropriate to a real (physical) wall.

To use WISE on a building representation like Figure 1, we would need to modify the specified dielectric properties of each material to adjust for the ''doubling'' of most of the walls. The run-time for WISE on this problem would be at least four times the run-time of a representation with ''logical walls.'' (The time complexity is $O(n^{b+1})$, where b is the number of bounces in the ray-tracing process and n is the number of walls in the building.) Thus, we seek to replace most of the multiple walls in the drawing by appropriate single logical walls.

Some sort of plane-sweep algorithm seems like a promising way to identify and coalesce nearby walls. One problem is to pin down exactly *what* sort of plane-sweep: we need to define what it means to be ''nearby'' and to ''coalesce'' two walls. Since we were not sure the output would be useful — significantly smaller, yet still detailed enough to capture important properties of the original building — we needed a quick-and-dirty implementation to let us explore various possibilities and see how good the results looked.

We developed our program in small pieces connected in a pipeline. The heart of the processing is performed by a 40-line AWK program that computes the transitive closure of the following relation on horizontal segments: segments S and T are ''nearby'' if the horizontal projections of S and T intersect and the vertical distance between S and T is at most h. An equivalence class under the transitive closure of ''nearby'' is a collection of line segments $s_1, \ldots, s_k$. Each such equivalence class is represented in the output by a line segment whose endpoints have *x*-coordinates equal to $\min\{ s_i.xmin \mid 1 \le i \le k \}$ and $\max\{ s_i.xmax \mid 1 \le i \le k \}$, and whose *y*-coordinate is $(\min_{1 \le i \le k} s_i.y + \max_{1 \le i \le k} s_i.y)/2$.

To compute the transitive closure of ‘‘nearby,’’ the program sweeps over a sequence of horizontal line segments that have been sorted by *y*-coordinate. At each segment *S*, it begins to grow a rectangular box. Initially, the top and bottom of the box coincide with *S*. As the program considers segments that lie later in the sequence than *S*, whenever it finds a segment *T* that lies ‘‘nearby’’ the top of the box, it ‘‘coalesces’’ *T* into the box: the vertical edges of the box are set to coincide with the horizontal extrema of the box and of *T*, the top of the box is set at the *y*-coordinate of *T*, and *T* is removed from the sequence of segments.

This implementation of plane-sweep will win no awards for asymptotic efficiency, but it led us to discover some interesting aspects of the problem:

- Some input line segments are not quite horizontal; we added a tolerance to accommodate this.
- Some input line segments almost touch at their endpoints; we added another tolerance that accounts for this.
- The picture of the building is still recognizable when only the horizontal line segments output by the coalescing algorithm are drawn.

Emboldened by the third discovery, we extended the coalescing process to the three other principal compass directions: positive and negative unit slope, and vertical. The easiest way to do this was to extract walls of these slopes from the original data, rotate them so they were horizontal, pass them through the original coalescing program, then rotate them back into their original positions.

Figure 3 shows the same floor of the Learning Center building as Figure 1, but after it has been processed by our program. (For this test, the *h* of the ‘‘nearby’’ relation was taken to be ½ foot, walls that did not lie very close to one of the principal compass directions were preserved unchanged from input to output, and features shorter than 1½ feet on a side were eliminated after the coalescing operation had been performed.) The drawing is clearly similar but much simpler. Equally clearly, places remain where the process has not simplified things as much as we might have hoped, for example, the residual drafting lines at the top.

One (coarse) measure of the quality of the coalescing step is the reduction in the number of walls. Table 1 shows some computational results for parts of three buildings.

It is difficult to quantify how well the output from our program resembles the original building. For some buildings, however, we can compare the output to the results of a human abstracting ‘‘significant’’ features from DXF input. Figure 4 shows the same building as extracted by a person working for five to ten hours. There are 1572 walls in the manually extracted version and 1291 in the automatically extracted version.

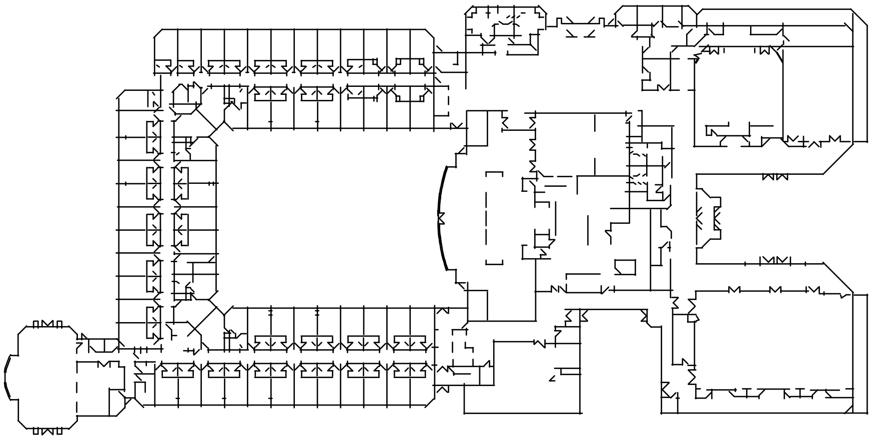


Fig. 3. Learning Center after wall coalescing

	Learning Center Floor 1	Middletown Floor 2	Red Hill Bldg 2 Floor 2
DXF source lines	691K	199K	89K
Total ‘‘walls’’	24109	9782	3744
Less non-walls	9144	9782	3744
After coalescing	1994	3051	951
Horizontal	660	1153	430
Vertical	756	1192	388
Slope ± 1	412	556	40
Other slanted	166	150	93
Less small features	1291	1657	487

Table 1. Some experimental results

The most important measure of the success of our program, of course, is how well it allows WISE to predict radio propagation. We have done limited experiments to compare propagation predictions on automatically and manually extracted drawings of the same building. To oversimplify somewhat, the general shape of coverage is similar, but until wall composition and ceiling location are set more carefully, detailed comparisons are inappropriate.

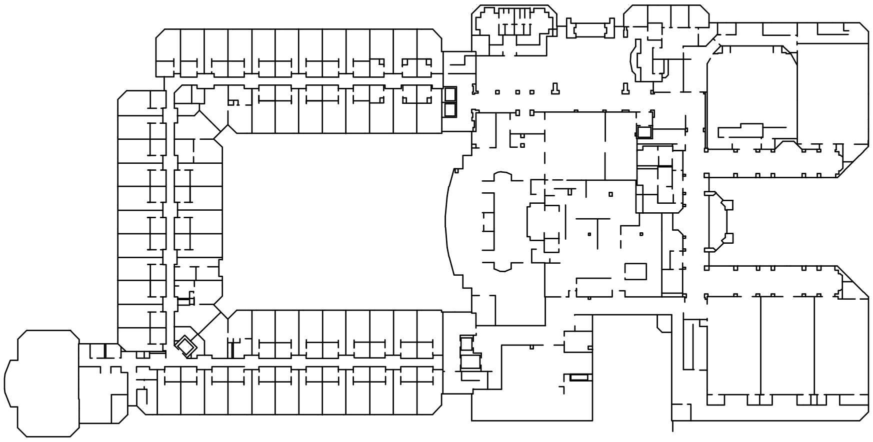


Fig. 4. Learning Center extracted by hand

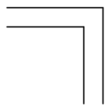
5 Remaining Problems

For the buildings we have examined, wall composition is not indicated in the DXF file. For now, we use color information to eliminate some constructs that are clearly not walls, then make educated guesses about what remains. In the Learning Center, for example, structural walls appear to be denoted by two colors — one for exterior, another for interior. We have called the former concrete and the latter sheetrock, which is typical of such buildings.

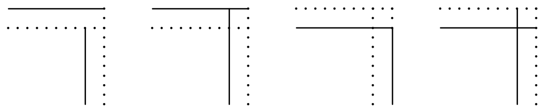
None of the DXF files that we have seen include useful *z* information. To permit a sensible WISE simulation, we have adopted the expedient of using the bounding box to synthesize a floor, a ceiling, and a plenum one meter below the ceiling, again typical of most office buildings. Some kind of alpha-hull might be a better choice than the bounding box as a false floor or ceiling, but there is no point getting carried away by analysis: it is difficult to imagine any automatic method that would recognize the open-air courtyard in the Learning Center.

One of the drawbacks of the sweep algorithm is that the resulting horizontal and vertical walls need not meet at a corner. For example, the input in Figure 5(a) can produce any of the outputs of Figure 5(b), among others; only 4 of the 16 possible outcomes produce a closed corner. These gaps make only a small difference to predictions about radio propagation: there is leakage through the gaps, but the number of such paths is small, since the default grid upon which coverage is computed is about 2½ meters on a side. The imperfect corners are visible even in Figure 3, however, so some subsequent cleanup might be desirable for purely cosmetic reasons.

We were fortunate to find building drawings in DXF format. Given only paper blueprints, we would need to scan them, use image-analysis techniques to discover the walls, then ‘‘beautify’’ the results [PVW85] to join corners, align walls, and the like.



(a) Input



(b) Some of the 16 possible outputs

Fig. 5. Possible results of merging process

([RMM+94] describes an alternative, bitmap-based approach.) An alternative is to enter significant points from the drawing using a digitizing tablet; experience suggests that this can be done for a building of modest size in a few hours per floor. The techniques we have described could be used in a variety of ways to aid digitization: they might be used to produce a ‘‘first draft’’ for subsequent refinement by hand; or they might be used to highlight possible errors after the user has completed a digitization by hand. Because the DXF files exhibit such idiosyncratic properties, some human intervention is unavoidable, but there are ways to use these techniques to help it along.

6 Conclusions

Our experience taught us again the importance of an experimental approach. Real problems are rarely specified in a neat, precise form, so it is essential to be able to try things. Only through experiments could we establish which parts of the problem matter, what assumptions seem valid, what will (or won't) work in practice, and what might be worth doing. We proved no general theorems about our techniques, although it was occasionally useful to think about what was causing some would-be theorems to fail.

The evolution of our treatment of small features offers an example of the value of experimentation. At first we discarded small features early in processing, to reduce the amount of data to be considered by later phases. Later, we noticed that many small features appear at corners, where their presence during coalescing could help to make walls meet properly. This led us to make removal of small features the final processing step.

As an example of thinking about theorems, an earlier version of our program swept over the plane, coalescing nearby segments only when both lay within a narrow sweep-window. This made the output dependent on the sweep direction, so the program did not compute a relation at all. Our final program is slightly more complicated, but it computes a well-defined mathematical function of its input. The resulting output is also about 10% smaller and looks better.

For ease of experimentation, it was invaluable to have a simple uniform representation of information. Fortunately, DXF, though complicated and clumsy, is well documented and represented in ASCII. We converted it to an even simpler format: one line per entity, each with a type, a serial number, its coordinates, and its parameters. This straightforward data representation meant we could easily use existing tools to sort, select, summarize, transform coordinates, and the like. We believe that a set of small tools that can be connected in a pipeline proved more useful for our exploratory experiments than would a library of computational geometry code that required us to program in C or C++.

For geometrical applications in two or three dimensions, it is essential to be able to see the results of computations. We used Tcl/Tk [O94] extensively. Tcl/Tk was originally intended for creating graphical user interfaces for the X window system, but its ''canvas widget'' is an extremely flexible and convenient tool for visualizing line drawings. For instance, a 25-line Tcl/Tk program implements a viewer that lets us zoom in and out and pan over building diagrams, identify any object with a mouse click, and hide or color any class of objects in the picture.

Simple algorithms and data structures — sorting, files, and one-dimensional arrays — sufficed for our exploratory work. Should wall extraction turn out to be sufficiently attractive that it will be heavily used in practice, then we might need a proper implementation of an efficient, robust plane-sweep algorithm.

Acknowledgments

We are grateful to Orlando Landron and Chris Read for providing DXF data for our experiments, and to Jon Bentley for helpful comments on this paper.

References

[AKW88] A. V. Aho, B. W. Kernighan, and P. J. Weinberger, *The AWK Programming Language*, Addison-Wesley, 1988.

[FGK+95] S. J. Fortune, D. M. Gay, B. W. Kernighan, O. Landron, R. A. Valenzuela, and M. H. Wright, ''WISE design of indoor wireless systems: practical computation and optimization,'' *IEEE Computational Science and Engineering* **2**, Spring, 1995, 58–68.

[F96] S, J. Fortune, ''A beam tracing algorithm for prediction of indoor radio propagation,'' these proceedings.

[MvR94] J. D. Murray and W. vanRyper, *Encyclopedia of Graphics File Formats*, O'Reilly, 1994.

[O94] J. K. Ousterhout, *Tcl and the Tk Toolkit*, Addison-Wesley, 1994.

[PVW85] T. Pavlidis and C. J. Van Wyk, ''An automatic beautifier for drawings and illustrations,'' *Computer Graphics: Proceedings of SIGGRAPH '85* **19**(3), 1985, 225–234.

[RMM+94] K. Ryall, J. Marks, M. Mazer, and S. Shieber, ''Semi-automatic delineation of regions in floor plans,'' Harvard University, 1994.