Interpreting Physical Sketches as Architectural Models

Barbara Cutler

Rensselaer Polytechnic Institute

Joshua Nasman

Rensselaer Polytechnic Institute

Abstract. *We present an algorithm for the automatic interpretation of a rough architectural sketch as a consistent 3D digital model. We compare our results to the designer's intended geometry. We further validate the algorithm by studying the variations in possible interpretations made by other humans for a set of relatively ambiguous sketches. In our system, the user sketches an architectural design by arranging small-scale physical wall modules and simple markers for windows on a table. These color-coded elements are captured by a camera mounted above the scene and recognized using computer vision techniques. The architectural design is automatically inferred from this rough physical sketch transforming it into a consistent and manifold 3D triangle mesh representation. The resulting digital model is amenable to numerous building simulations including lighting, acoustics, heating/cooling, and structural analysis.*

1 Introduction

Sketching, drawing, and diagramming are fundamental components in architectural design. The evolution, communication, and documentation of a design are performed through various styles of visualization. These broad categories of visual communication in architectural design use a variety of artistic techniques; however, these representations are usually stylized and employ common conventions.

One important category of architectural illustration is the *figure-ground diagram*. This visual representation uses two contrasting colors, positive and negative, to partition space into two sets by filling in large regions of the diagram with solid color. In architecture, a figure-ground diagram is most often drawn in plan (from above) to convey either the rough overall massing shape of the building or the public freespace; for example, a public plaza surrounded by private buildings. Often architects will execute diagrams in both forms when considering different aspects of the same project. Another important class of diagrams used in the early stages of architectural design, *circulation diagrams*, visualizes how people will use the space and highlights the common movement paths within the proposed design. By analyzing and anticipating common paths, the relative placement of spaces within a design and the relationship to the existing site can be optimized to minimize path lengths or to add interest or drama by highlighting views and enhancing the experience of circulating within the design.

Figure-ground and circulation diagrams are used primarily in the early stages of design when the spaces and relationships are still evolving. In contrast, technical architectural CAD drawing, used in the later stages of design and in construction documentation, is highly precise and detail-oriented and more strictly follows diagrammatic conventions. Few architects would claim that traditional CAD modeling tools and detailed technical drawings are essential to the early, creative stages of architectural design.

In addition to pen-and-paper sketches, small-scale physical 3D models (often built from scrap cardboard) are fundamental tools for architectural design. These study models can be essential for understanding complex spatial relationships, documenting the evolution of a design, and communicating the concept to the client. Even with the wholehearted adoption of computer technology for drafting and 3D visualization, the physical study model has not been abandoned as a tool for architectural design. In fact, rapid prototyping technology has increased the expectations for physical prototypes of complex designs.

1.1 Tangible User Interface for Architectural Design

The architectural modeling system at the center of our project uses a *tangible user interface*, which involves manipulation of physical props for interaction with computation (e.g., [Ben-Joseph et al. 2001]) rather than the typical mouse/keyboard/monitor interaction between human and computer. Well-designed tangible interfaces are attractive because they are inherently simple, natural, and intuitive. Furthermore, these interfaces generally support collaborative work environments.

In our system, shown in Figure 1, one or more users gather around a table and construct a small-scale (1:12, 1” = 1’) sketch of an architectural design using simple foam board flat and curved walls in three different heights (5”, 8”, and 10”). Special markers slip over the top edges of the walls to indicate windows, and the overall orientation of the architectural design on the site is specified with a “north arrow” token. This design environment is simple to operate and requires essentially no instruction to use. The only restriction on the designs is that wall elements must be upright, resting on small “feet”, so that each wall surface is perpendicular to the table surface. A new design can be quickly constructed in under a minute by selecting and arranging wall and window elements from a modest collection of parts on a neighboring table. Similarly, the design can be edited in seconds by adjusting any of the physical pieces. Image capture and processing of the detected geometry is completed in a couple of seconds. The system supports viewing and editing by multiple users who are gathered around the table. The interactive modeling environment encourages creativity and collaboration.

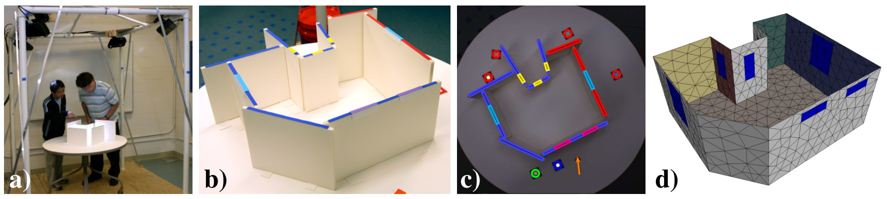

Figure 1: In our physical architectural design environment a) users gather around a table and construct b) a small-scale mockup of a design from a collection of wall elements and marker tokens. A camera above the table c) captures the layout of elements on the table. Our sketch interpretation algorithm processes these elements to construct a consistent and watertight triangular mesh of the implied architectural design (ceiling removed for visualization).

Within this design environment the designer cannot create a highly detailed model, but instead is compelled to focus on more abstract concepts appropriate for the early stages of design, including orientation of the building on the site and spatial relationships between the primary zones of the design, and in making key decisions about the structure, lighting, and acoustics. Importantly, computational simulations to analyze the performance of the structure, lighting, or acoustics of the space (which are currently underutilized during schematic design) require a consistent and watertight, yet simple, 3D representation of the design for efficient and accurate analysis. Thus, we believe our algorithms to produce such a model from these sketches will be an invaluable tool in the early stages of architectural design.

1.2 Contributions

In this paper we present the following contributions:

- An algorithm and implementation for the automated interpretation of physical sketches as consistent architectural designs which can be exported as either standard architectural floor plans or watertight 3D triangular meshes amenable for simulation.
- The collection of several hundred physical architectural sketches using our design environment. Each design is annotated by the original designer to indicate the intended interpretation of the design.
- Re-interpretation of these designs by other humans to provide a measure of the ambiguity present in these sketches. The set of human interpretations is compared to the algorithm's output for validation.

1.3 Overview

We summarize several important areas of related work in Section 2. In Section 3 we present our sketch interpretation algorithm, which was developed with extensive user testing and feedback from both architects and non-architects. In Section 4 we present the results of two formal user studies that we conducted to gather a large set of example designs and validate our sketch interpretation approach.

2 Related Work

Our project draws from research in a wide variety of areas including: sketching interfaces, sketch recognition, human perception, and computational models of gestalt and saliency. In the sections below we provide a brief overview of prior art in these fields and existing software for architectural design.

2.1 Modeling Interfaces for Architectural Design

Many existing computer software packages tackle the challenge of constructing 3D architectural geometry. Initially these packages focused almost exclusively on creating models with high geometric precision, requiring a significant investment of time and only limited support for editing a completed model. Unfortunately, because focusing on precision too early in the design process can stifle creativity [Lawson 2002], these tools are generally used only in the later stages of design.

The requirements for 3D models in architecture vary tremendously depending on the intended use of the model. Photorealistic renderings require high polygon count geometry and accurate normals and materials. In contrast, many simulations (e.g., acoustic, passive ventilation, heating/cooling, structural analysis) require a consistent and watertight geometric model for accurate calculations and often perform most efficiently with a simplified model constructed especially for that analysis [Autodesk 2000-2008].

More recently, software tools have become more amenable to the fast-paced, quick sketching of the early stages of design [Google 2010], including explorations of pen-based user interfaces for 3D modeling [Zeleznik et al. 1996; Igarashi et al. 1999; Lipson et al. 2002]. A limited construction interface (axis-aligned elements on a floor plan grid) can help ensure the construction of architectural environments that are appropriate for use in virtual reality walkthroughs [Mackie et al. 2004]. New drawing user interfaces and systems have been demonstrated that allow architects to leverage their pen and paper skills when interacting with the new media interface of the computer. Using projective geometry, freehand architectural sketches can be re-projected or warped to simulate novel viewpoints and an immersive experience [Tolba et al. 2001]. The Mental Canvas system allows designers to arrange 2D sketches on arbitrary planes in 3D, constructing an effective representation of complex architectural spaces [Dorsey et al. 2007].

2.2 Sketch Recognition

Sketch recognition systems are typically custom-developed for each application (e.g., recognizing hand drawn UML diagrams [Lank et al. 2000]) and leverage domain-specific knowledge to improve accuracy. General-purpose toolkits and languages for sketch recognition of diagram components can ease the development of these programs [Hammond and Davis 2007].

Well-designed sketching user interfaces minimize the number of times the user is prompted for additional information or design annotation, which interrupts workflow and concentration. Strategies for continuous and incremental recognition of drawings as they develop have been demonstrated [Alvarado and Davis 2001]. It is important to maintain an estimate of the confidence in the intermediate interpretations, which improves the accuracy of the final interpretation. An ongoing challenge in sketch recognition is the grouping of multiple individual strokes that form a single logical unit. The drawing sequence for individual strokes and other meta-data available from tablet displays [Wacom 2010] can be used as evidence when tackling this problem [Gross 1994]. Another challenge in sketch recognition comes from touch-up or continuation strokes the user might make to complete a drawing [Paulson 2010]. These strokes are disconnected from or overlap other strokes but are intended to be recognized as a single object. Spatially close strokes (often defined as the distance between two endpoints being less than 10% of their average length) can be merged; however this greedy approach, while fast enough for interactive sketching, is not optimal.

2.3 Sketch Recognition for Architectural Drawings

The precision, consistency, and standardization of architectural CAD drawings make this domain amenable to automatic processing algorithms, and an impressive collection of automated sketch recognition programs have been developed for architectural drawings [Ah-soon and Tombre 1997; Ah-Soon and Tombre 2001; Kulikov 2004; Lu et al. 2005]. The motivation behind many of these tools is the digitization and automated reconstruction of 3D building geometry from older construction documents. These methods have also been demonstrated for hand-drawn architectural designs that closely follow the diagrammatic conventions of CAD drawings [Aoki et al. 1996; Llados et al. 1997]. For example, the VR Sketchpad project automatically interprets a 2D floorplan sketch including furniture layout to create a 3D VRML model that can be used for walkthroughs [luen Do 2001]. Freehand sketching can be used to interact with digital models [Gross and luen Do 2000] and preliminary work was done to classify interior and exterior walls in quick floorplan sketches [Ramagupta and Hammond 2007].

Koutamanis and Mitossi describe three levels of automated recognition of architectural floor plans: recognition of geometric elements, recognition of building elements, and recognition of spatial articulation [Koutamanis and Mitossi 1993]. They argue that the third category is the most advanced: identifying solid mass versus space within the design. Our aim in this work is to specifically address this challenge for freeform architectural sketches using a tangible interface.

2.4 Gestalt Theory and Sketch Interpretation

Gestalt psychology, the laws of perceptual organization, and Pragnanz [Koffka 1935; Kanizsa 1979] describe how humans perceive and interpret incomplete diagrams or other modes of partial stimuli. The fundamental phenomena of closure, proximity, symmetry, and continuity can be explained by low level human vision processing. Gestalt theory describes why our interpretation of an incomplete or ambiguous diagram tends toward simpler forms, avoiding complexity. The rich vocabulary of pen and paper sketching in architectural design draws on the gestalt principles of collinearity, parallelism, continuation, and completion [Koutamanis 1999]. Our algorithm for automated interpretation of architectural sketches follows and implements these principles. Our user studies on human interpretation of architectural sketches provides validation to our proposed use of these concepts in recognition and analysis of architectural designs.

Research in computer vision has developed algorithms for image processing using computational models of gestalt. Attributes which define the form, i.e., thickness of a line, convexity, and parallelism, can be referred to as *gestalts* [Desolneux et al. 2002]. Computational gestalt research focuses on determining thresholds that indicate a pattern is significant. In other words, this work involves detecting and studying patterns and analyzing how likely those patterns would be to occur in the image randomly [Desolneux et al. 2002].

Similarly, saliency is a measure of how much an area stands out in comparison to the areas around it. A *saliency map* [Koch and Ullman 1985] combines different stimuli (movement, color, etc.) and the relative conspicuity to quantify changes in these characteristics. Itti, Koch and Niebur designed a computational visual attention model based on the primate visual system to estimate the saliency map and identify which areas of the scene are most likely to contain useful information and should be analyzed in more detail [Itti et al. 1998].

3 Sketch Interpretation Algorithm

In our physical sketching environment, the user selects from the provided collection of physical props and quickly assembles the chosen pieces on the table. Thus, the arrangement of elements truly forms a rough, approximate sketch. The selected wall pieces are likely too long or too short, yielding overlaps or gaps with adjacent walls. Similarly, the limited palette of curved components may not have the desired curvature and thus tangential connections are imprecise. Furthermore, the approximate assembly manner means that components intended to be parallel or meet at crisp 90° angles will include some unavoidable imprecision. The challenge is to sift through the available information to deduce and construct the clean and complete design as it was conceived in the architect's mind.

In the following sections we present the key steps of our sketch interpretation algorithm: image pre-processing, detecting parallel, perpendicular, and collinear elements, linking elements into chains, constructing an arrangement of polygonal cells, estimating spatial enclosure, assigning interior/exterior zones, managing details, and post-processing the floorplan geometry to make a 3D model.

3.1 Image Pre-Processing

A camera mounted above the table captures the details of the physical sketch. A controlled lighting environment and carefully chosen color-coded labels on the physical

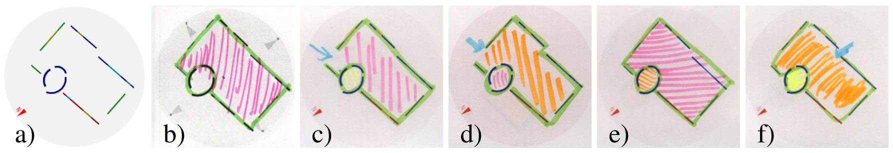

Figure 2: Intended collinearity can be ambiguous: a) detected primitives, b) annotation by the original designer, and c-f) annotation by other users.

elements allow this geometry to be robustly detected using standard image processing techniques, which are described in detail in our earlier publications [Sheng et al. 2009b; Sheng et al. 2009a].

The input to our sketch interpretation algorithm is the 2D projection of each detected wall module onto the table surface, labeled with the element height. The flat walls are rectangles and include any associated windows as inscribed rectangles. The circular arc curved walls are specified by a center, the inner and outer radii, and the start and end angles of the arc. An example of this input is shown in Figure 2a.

3.2 Intended Parallel, Perpendicular, and Collinear Elements

For practical construction and space efficiency reasons, most real-world architecture involves parallel walls and sharp 90° corners. Even within high-profile showpiece architecture, spaces are typically arranged into regular patterns and placed on a grid, sometimes with secondary grids that are offset and/or rotated. Many (but not all) designs created by participants using our tangible sketching interface follow these conventions and we can automatically detect the implied grid(s).

First, we cluster the flat walls into groups that are nearly parallel and *snap* all walls in each group to their weighted average direction. Similarly, we compare the wall groups to each other and those that are approximately perpendicular are snapped to be orthogonal. We found that an angle tolerance of 5° was an appropriate threshold across all of our collected design data. This tolerance was effective at cleaning up placement imprecision inherent in the physical sketching environment, yet was not large enough to introduce artifacts in the more freeform designs that eschewed parallelism and right angles.

In addition to the angle tolerance, it is necessary to identify flat walls that are approximately collinear. However, selecting a global setting for this offset tolerance distance is somewhat more difficult. We also saw variable tolerances for collinearity in human perception when different users were asked to interpret the same design (Figure 2). Some users perceived a slight line break as an intended straight line, while others interpreted the break as an architectural feature and possibly an entrance. We found that using a 1” offset distance tolerance in our physical sketching environment (equal to 1’ in full-scale) was a good compromise for our automatic collinearity detection and adjustment, but the user may need control of this threshold for some designs.

We have not yet finished implementation of similar clustering and adjustment methods for circular arcs. The implementation is straightforward and will require determination of appropriate tolerances and smoothing procedures for several cases: multiple arcs placed to sketch a circle or ellipse (e.g., Figure 2), two arcs that form an inflection point, and adjustment of the tangent and/or curvature when a circular arc leads into a flat, straight wall.

3.3 Linking Walls to Form Continuous Chains

Following the gestalt principle of continuity we not only snap nearly collinear elements to a common line, but we also form explicit connections between elements with similar tangents (straight to straight, straight to curved, or curved to curved). Two or more elements can be connected into a *chain* that is traced through the working plane, separating space into two regions. Our angular and offset distance tolerances for establishing a connection between two elements is less strict than the parallel and collinear thresholds described in the previous section, allowing the establishment of longer freeform chains. Examples of these chains are shown in Figure 3c.

Our algorithm for establishing the chains is as follows. For each endpoint of each wall element we search all other walls for the best matching connection tangent. If a pairwise connection is mutual (element A selects element B as the best match for tangent direction and offset distance and element B also selects element A as the best match), then the connection is established. If no connection is made for a wall endpoint, then that end of the wall chain is simply extended to infinity following the tangent of the wall at that endpoint. When a wall chain connects curved arcs, the chain may form a U turn (Figure 3 second row), a closed loop (Figure 3 third row), or other interesting shapes.

Defined spaces in architecture can be created by real and/or implied boundaries. Each wall chain divides the working plane into two spaces, one on either “side” of the chain. The set of all wall chains in a model will divide space into many subspaces that can be labeled by their sidedness, which is visualized in Figure 3d. If a wall chain loops around and crosses itself (Figure 3 fourth row, blue and yellow chains), the loop portion of the chain is disconnected to define a new chain to allow the unique labeling of all subspaces. Note that if two wall chains cross two or more times (which is possible if one or both chains are non-linear), the resulting subspace organization may have two disconnected spaces that have the same sidedness (Figure 3 fifth row, blue and red cells). We perform a connected component analysis to identify this situation and define separate subspaces.

3.4 Arrangement of Cells and Enclosure

The wall chains described in the previous section are used to cut the working plane into a watertight planar convex polygonal mesh or arrangement of cells, which is represented using a half-edge data structure. Each wall chain explicitly represents the wall thickness, thus working plane polygons are constructed for both open space (interiors and exteriors) as well as the are comprising the construction thickness of real and implied walls.

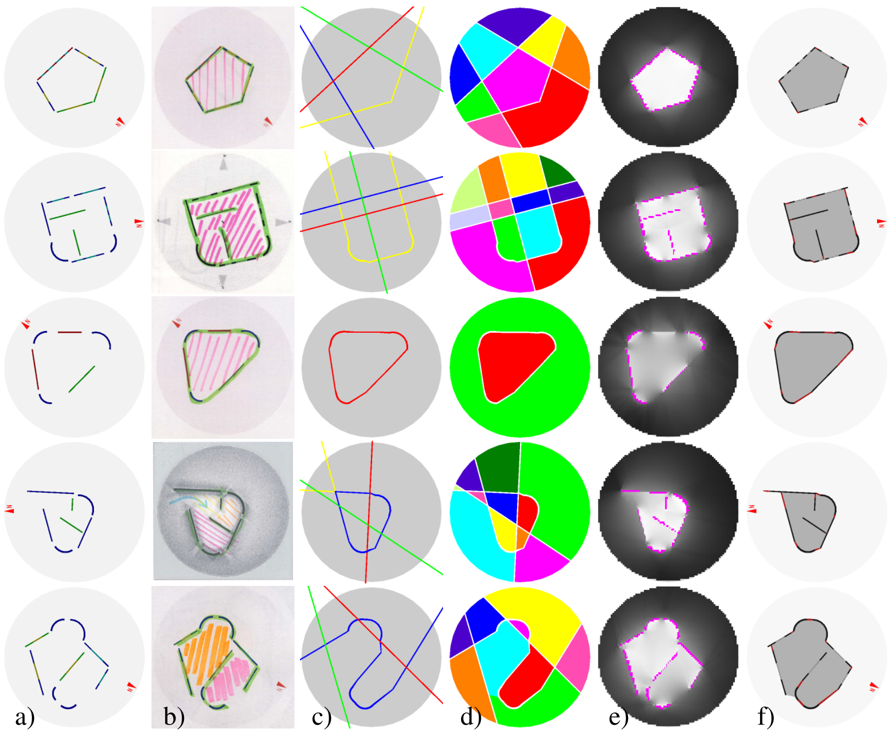

Figure 3: Dividing space into cells by extending tangents and connecting elements: a) detected primitives, b) annotation by original designer, c) wall chains, d) zones defined by the wall chains, e) enclosure, and f) our automatic interpretation.

For each cell in the arrangement, we calculate the *enclosure*, the probability that the cell is part of the building interior. We define the enclosure at a point as the lack of visibility to areas outside of the design from that point. We estimate this value by tracing rays from the point to infinity in a dense sampling of directions and record the fraction of all rays that intersect a wall of the design. We visualize enclosure over a dense point grid in Figure 3d. Points with high enclosure (nearly all rays intersect a wall in the design) are drawn light grey or white, and points with low enclosure are dark grey. We define the average enclosure of the cell in the arrangement by averaging the enclosure at many points within the cell. Similarly, we can compute the average and standard deviation of the point-based enclosure values for a subspace.

For relatively simple designs, a carefully-selected global threshold placed on the point/cell/subspace enclosure value can correctly classify each subspace as interior or exterior. However, as the gaps between the walls increase or decrease the threshold value must be adjusted accordingly (compare the first and third rows of Figure 3d). Furthermore, if the model contains concavities in the outer wall, nearby

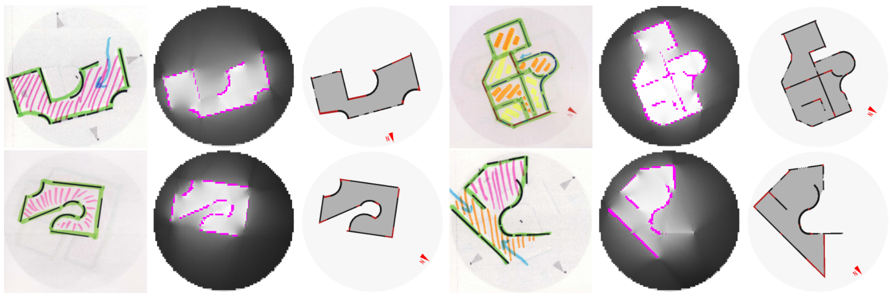
Figure 4: Designs with non convex boundaries may not be accurately extracted with a simple threshold on the average point or cell enclosure. By minimizing the lengths of unused wall and extra *inferred wall* necessary to enclose the interior zones we correctly interpret these complex designs.

areas may be incorrectly marked as interior spaces. Beyond the challenge of selecting a threshold, for more complex models a simplistic setting of a global threshold will not successfully separate the building interior from exterior (Figure 4).

One cause for incorrect interior/exterior division is high variance in enclosure within a single subspace. For example, consider the final row of Figure 3. The sharp bends of the blue wall chain loop through the design and interact with the other two wall chains to produce one large subspace, colored cyan and green. The enclosure values within this subspace have high variance, indicating that it should not be treated as a single space when identifying interior space. For each subspace with high enclosure variance, we solve a Minimum Cut graph problem to determine an optimal segmentation of this subspace. We build a dual graph from the polygonal cells in this subspace. Each polygonal cell in the arrangement is a node in the graph. If two cells share an edge in the arrangement, we create an edge between the corresponding nodes in the graph. The weight of the edge is defined to be the length of the shared edge in the arrangement. The source is defined to be the node whose corresponding cell has the highest enclosure, and the sink is likewise defined to be the cell with the lowest enclosure value. Using a basic textbook implementation of the Maximum Flow/Minimum Cut algorithm, we find a minimum length cut which divides the zone into two subspaces that will have lower variance and produce a more satisfactory interior/exterior segmentation of the design.

3.5 Assigning Interior and Exterior Zones

After the wall chains have divided the working plane into a set of zones, we need to label each zone as either an interior or exterior space. The average enclosure value for the zone can be used to make an initial determination, but that strategy frequently yields incorrect assignments for complex designs with non-convex boundaries (e.g., Figure 5). Furthermore, our method for constructing wall chains that extend each element to infinity will yield a division of space into zones far from the element's

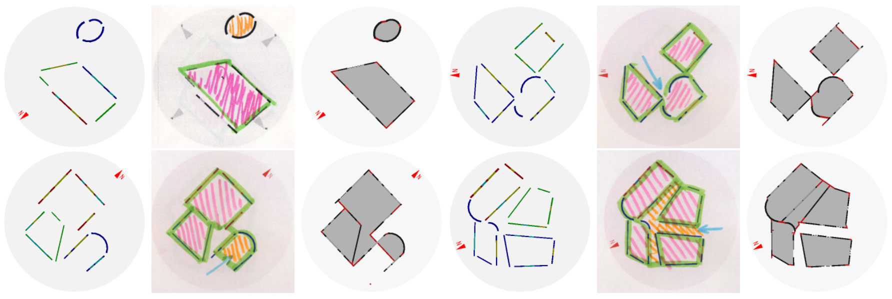

Figure 5: Challenging examples of designs with multiple rooms. The examples in the bottom row are somewhat ambiguous and have multiple reasonable interpretations for the passageways between rooms.

actual position, which may not be intended. When two neighboring zones are assigned opposite labels, the wall or wall chain between the zone is thus interpreted as an implied wall or boundary. If the original wall element is short and/or a significant distance from the implied boundary, the resulting automatically-constructed floor plan may be non intuitive and not match a human interpretation. Therefore, we must be careful to use an extended wall chain as evidence for a boundary only in close proximity to the original wall. Every hypothesized or inferred wall separating interior and exterior space should be checked against the evidence.

In solving this problem, we follow the Gestalt principles of closure and simplicity. We search for a closed form that is simple, uses most or all of the detected wall elements, and requires little length of additional inferred exterior wall to fill gaps between the original wall primitives of the sketch. We solve this optimization problem in a brute force manner by considering as interior space all subsets of the zones with enclosure values above a reasonable threshold and select the zone assignment that minimizes the sum of all unused walls and all inferred walls. An unused wall is defined as a detected element that has exterior zone on both sides. Note that wall elements may be partially used and the unused wall penalty is accordingly prorated. Similarly, an inferred wall is a portion of wall chain that is not represented in the sketch by a physical wall, but has exterior zone on one side and interior zone on the other. In the floor plan results diagrams used throughout this paper, real walls are drawn in black, interior zones are drawn in medium grey, exterior zones are white and inferred walls are drawn in dark red for visualization purposes.

Some of the collected designs contain an interior space that might have been conceived as an open courtyard rather than a room with no exterior walls (Figure 6). This distinction can be important for architectural simulations such as daylighting design or passive ventilation — whether the finished design includes a roof over this fully interior room will impact the simulated performance results. To resolve this ambiguity, we are considering user interface options for allowing the designer to indicate which of these two perfectly reasonable scenarios he/she intended. Importantly, we want maintain our minimalist interface and follow the most appropriate default interpretation.

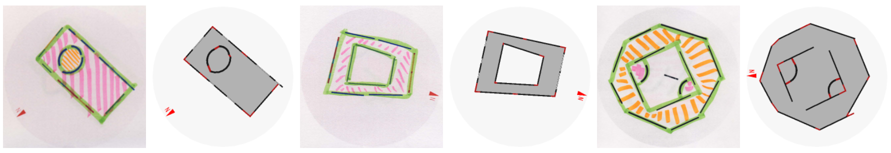

Figure 6: If the design contains a interior room with no exterior walls, this space may be intended as a courtyard space, uncovered by a roof. The system will require extra information from the designer if the default interpretation does not match his/her intentions.

3.6 Detecting Inferred Interior Walls and Trimming Unused Walls

Once the core interior/exterior spaces of the design have been determined and the true exterior and inferred exterior walls have been identified, the system performs several small cleanup steps to improve the quality of the final model and ensure that the extracted 3D geometry is manifold. When two wall chains that each define a portion of the boundary between interior and exterior cross, the quadrilateral zone defined by the thickness of each chain at the crossing should also be labeled as a wall to create proper interior and exterior corners.

Some designs incorporate interior partition walls that are meant to separate the interior into smaller rooms. Due to the physical sketching environment, generally these walls are a bit short and leave small gaps where they should meet other interior or exterior walls. However, not all gaps should be closed. Standard doorways in architecture are generally a minimum of 2’6” wide, so if the gap is less than 2” wide, we assume this was intended to be a solid wall and we close the gap by labeling the corresponding wall chain polygons as inferred walls. We found that this tolerance provided a reasonable match to the designer’s annotation of his/her intentions for the full range of example designs. Similarly, when the physical wall modules are slightly too long and protrude from the middle of an inferred boundary or corner, we should clip back this geometry to create a more polished design. We propose a tolerance of 1” (corresponding to 1’ in full scale) for this trimming. We note that it is important to not completely remove from the design all walls that are “unused” (not positioned on the exterior/interior boundary of the model or used as interior partitions). In many cases these extra exterior walls serve specific architectural functions including privacy screens, shading, acoustic dampening, and wind control.

Finally, we propose an initial strategy for labeling the primary entrance to a design, and augmenting the floorplan and 3D model with this information. Note that not all designs have a clear entrance, but many of our user study participants left specific openings within the outer boundary of their shape. Simply detecting the longest section of inferred exterior boundary (and greater than a minimum tolerance of approximately 4”) as the primary entrance will correctly label this feature in most of the designs. One notable example that breaks this rule is shown in the bottom row of Figure 7. Annotations made by other user study participants match the intention of the original designer: the obvious entrance is through an elaborate portico, rather than through one of the large gaping holes in the “back” of the de-

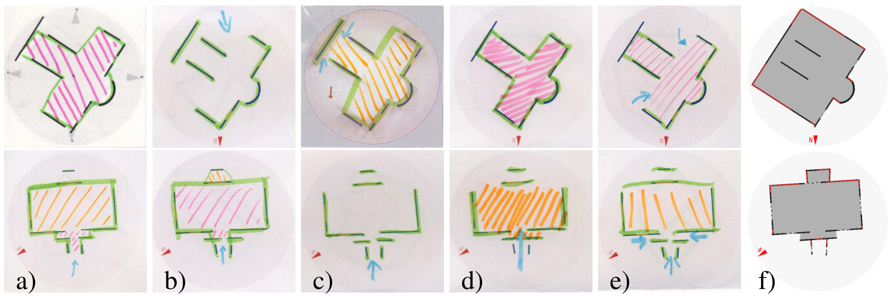

Figure 7: Domain-specific knowledge may be necessary to correctly interpret sketches that hint at common architectural forms, such as the cruciform used in church floor plans (top row) or to recognize an entrance portico (bottom row). Despite the potential for ambiguity, b-e) most users' interpretations matched a) the original designer's intention. Our automatic sketch interpretation results are shown in f).

sign. This sketch interpretation task will benefit from domain-specific knowledge of common architectural forms.

3.7 Post-Processing: Constructing a Watertight Triangle Mesh

At the end of our sketch interpretation algorithm, we have a precise, manifold, polygonal representation of the working plane. Each polygonal cell has well defined neighboring cells (no 'T' junctions) and has been assigned one of several labels: *solid wall*, *window within a wall*, *interior space*, or *exterior space*. This representation can be extruded and exported as a consistent mesh following the necessary CAD conventions for architectural rendering or performance simulation software. For example, we can construct a watertight mesh appropriate for a radiosity simulation of interior illumination as follows. Each interior cell is exported as two polygons, one floor plane polygon with normal pointing 'up' and one ceiling polygon with normal pointing 'down'. For every edge between an interior cell and a wall or exterior cell, we create a wall polygon stretching from the floor to the ceiling with normal pointing toward the interior cell. When an interior cell touches a window cell, the exported wall polygon is split vertically and assigned different materials as appropriate.

4 Validation of Physical Sketch Interpretation Environment

To validate our algorithm for sketch interpretation, we ran two user studies, one focusing on our physical sketching environment and the second on sketch interpretation. We recruited participants with a variety of backgrounds including architecture, visual arts, and computer science.

4.1 Design Collection Study

The purpose of the first user study was to sample the range of architectural designs that could be constructed in our physical sketching environment and to evaluate the potential for its use in the early stages of architectural design. The design task was open-ended and after a brief overview of the tangible interface, participants were instructed to use the wall and window primitives to create between 10 and 20 different designs.

After each participant completed the design stage, we prepared a single-page annotation form for each of his/her designs. The form contained two parts: *designer annotation* and *evaluation of automated sketch interpretation algorithm*. The form was folded in half so that only the annotation section was visible and the participant was instructed to first complete the annotation for all designs before unfolding the paper to see the output from our automated sketch interpretation algorithm. Thus, the participants were not influenced by the output of our program in either the design or annotation stages.

The annotation portion of the form presented two large images: the overhead photograph of the physical sketching environment (for reference) and a 2D rendering of the detected wall geometry (to be used for annotation). The participant was instructed to use a green highlighter to draw the complete intended wall geometry on the detected geometry rendering. The pink, orange, and yellow highlighters were used to shade interior spaces. Optionally, he/she could use a blue arrow to indicate an entrance or to sketch the circulation within the design. As guidance, users were provided with three example designs annotated in this manner.

The evaluation portion of the paper contained our automatically generated floor plan of the design. The users were asked to evaluate the quality of the automatic interpretation of each design, whether it matched the design intention, was an acceptable alternate interpretation, or was incorrect. Additionally, we encouraged them to mark or comment on which parts of the design were most challenging for the automated system to interpret. After completing the evaluation of all designs, the users filled out a short post-study questionnaire.

Each participant used our physical sketching environment for approximately 20 minutes and created 3-26 designs. Some users created just a few highly detailed designs, while others created many rough sketches or a series of variations that evolved from a base sketch. In total we collected 329 designs from 30 participants in the first user study. Fifteen of those participants (responsible for 154 of the designs) were architecture students, most with at least three years of formal architectural education and professional experience through internships. Of the other participants, eight were visual arts students (83 designs) and the remaining seven (92 designs) had no formal training in architecture or art. A broad selection of these designs are presented in Figure 9.

4.2 Re-Interpretation Study to Quantify Design Ambiguities

For the second study, we wanted to understand any discrepancies between our algorithm's interpretation and the original designer's intentions. We wanted to investi-

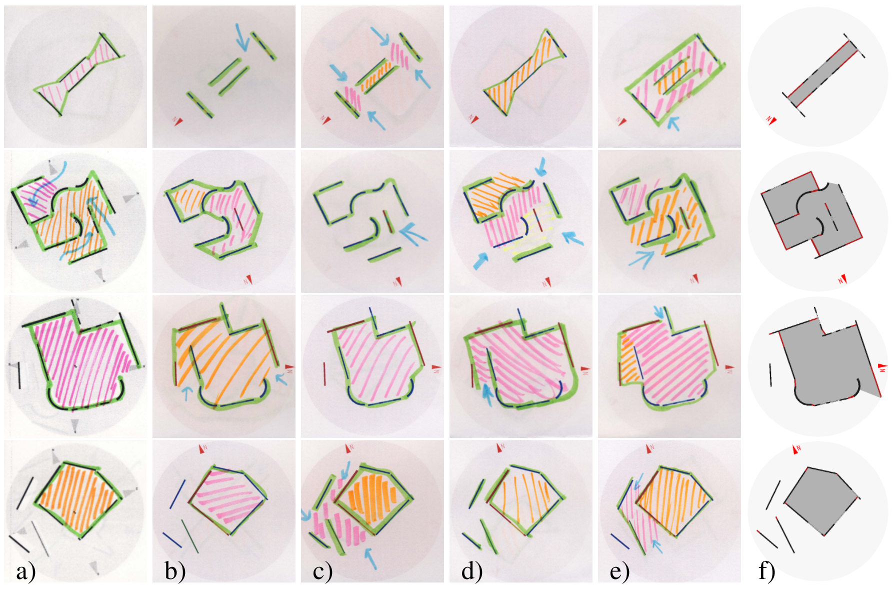

Figure 8: Some ambiguous designs: a) the original designer's annotation, b-e) annotation by other users, f) our automatic sketch interpretation results.

gate whether the differences were due to a flawed automatic interpretation strategy or the result of ambiguous physical sketches (Figure 8). In order to quantify the ambiguity of a particular design, in our second user study we asked the participants to annotate a selection of interesting sketches made by other users. All participants for the second study first performed the tasks of the first study (if they were not already subjects in that study) and thus all were familiar with the sketching environment and annotation instructions.

All designs from the first study that our initial sketch interpretation program struggled with were selected (omitting near duplicates), as well as other designs that we thought were ambiguous, complex, or interesting. We also included a number of simpler designs, which had a single reasonable interpretation, as controls. In total, 114 of the 329 total designs from the first study were selected. 60 of these designs were created by architecture students, 28 were made by visual arts students, and 26 were from students with no formal training in architecture or visual arts.

Each participant for the second study was presented with annotation paperwork for a randomly ordered, randomly selected subset of these designs. The annotation form consisted of three parts: *annotation*, *comparison to the original designer's intention*, and *evaluation of automatic interpretation*. The forms were folded to conceal the second and third parts.

As in the first study, the participants were asked to mark their interpretation of each sketched design and shade the interior spaces. After approximately 20 minutes each participant was asked to proceed to the second part of the study and any designs he/she had not yet annotated were collected.

Next, the participant was asked to unfold the paper (keeping the third part concealed) and compare his/her interpretation of each design to the original designer's intention, marking whether the interpretations matched or how the designer's physical sketch was ambiguous or unclear. Finally, after all comparisons were made, the participant unfolded the third and final part of the form and evaluated our computer algorithm for automated sketch interpretation.

We collected a total of 346 new annotations from 15 participants (124 annotations were made by architecture students, 82 by visual arts students, and 140 by other students). Each of the 114 designs received between 3-6 new annotations.

4.3 Validation Results: Subjective Feedback

The users' questionnaires revealed how excited they were about our physical sketching environment. A visual arts student said the system was "Very intuitive, very clear. Felt like playing with blocks as a kid, but each block had a meaning. Seeing each design in 3-D helped spike the creativity." Another visual arts student commented: "I was really impressed with the software–it did a great job mapping what I wanted." A second year architecture student said of the system "I was very surprised by the accuracy of the program for the most part. Despite some errors, the interior and exterior implied spaces were read pretty well." Other users were surprised at particular failings for rules we had not yet implemented. "The program filled in a void that was meant to be exterior, especially since I had windows on the exterior parts of these walls to make that distinction."

We used direct feedback from architecture students in pilot studies as well as general observations about the designs they created to improve our sketching environment and automatic interpretation algorithm. These improvements include: addition of curved walls and column primitives, control over window placement, detection of disjoint spaces and interior courtyards, and handling designs with large gaps in the exterior wall (typically, an implied entrance). We are motivated to continue this avenue of work and tackle the challenges of detection and labeling of the different subspaces within the interior, predicting interesting circulation patterns, etc.

The results of our second study can best by summed up by one student. Interpretation was "Often challenging. Many designs were unclear, difficult to interpret. Others were extremely clear and easy." Some users were surprised by the variety and complexity of designs possible in the system. A visual arts student said "I was surprised at the designs that the other users came up with – they seemed very complex in some cases – and the computer did a good job of interpreting them."

We found that for many of the designs that our algorithm struggled to interpret, other humans also found the design to be ambiguous. However, there were several notable examples where all humans interpreted the design quite similarly to the original designer, despite a lack of hard evidence for that shape within the sketch. Figure 7 presents a few of these examples, where the humans are quite consistent in their interpretation of the design. We believe this may be due to domain-specific

	correct		mostly correct		incorrect		total
clear	155	78%	17	9%	26	13%	198
ambiguous	74	56%	35	27%	22	17%	131
total	229	70%	52	15%	48	15%	329

Table 1: Statistics about the ambiguity of the physical design sketches and the quality of the interpretation results from our automated sketch recognition algorithm.

knowledge of architectural forms that have not yet been explicitly encoded in our sketch interpretation algorithm. One architecture student noted for the example shown in the top row of Figure 7: "Humans recognize this because it is a basic cruciform shape, but without more information, it may be difficult for an algorithm to determine this."

4.4 Quantitative Sketch Interpretation Results

After all of the designs and annotation figures had been collected, each physical sketch was categorized as "clear, straightforward, unique interpretation" or "ambiguous, multiple interpretations possible". This was determined by criteria such as if there were openings that may or may not be doors and whether it was clear which spaces were interior versus exterior. Secondly, each automated interior/exterior space partitioning by our algorithm was graded on the following scale: "correct", "mostly correct", or "incorrect". In order for a design to be marked correct, all interior spaces had to match and all walls that were part of the design had to match exactly with the designer's intention. An interpretation was judged to be mostly correct if at least 90% of the walls matched and if each interior space was mostly bounded by real walls. The results are shown in Table 1. In total, our algorithm found a correct interpretation for 70% of all designs and correct or mostly correct interpretation for 85% of all designs. Of the designs that were judged to have single clear interpretation, we made the correct interpretation for 78% of the designs. In contrast, for the ambiguous designs we found a correct interpretation (closely matching the original designer's intention) for 56% of the designs. Many of the errors in interpretation made by the system are minor robustness issues and we are confident that with additional development efforts the accuracy of the system will improve.

We analyzed the annotations to determine if there was any correlation between the architectural or visual arts training (or lack thereof) of the original designer or secondary annotators. We did not find a correlation between the background of the participant and their ability to correctly infer the original designer's intention.

5 Conclusion and Future Work

We have presented an algorithm for automatically interpreting approximate physical sketches of architectural designs, preparing detailed floor plans of these designs with explicitly represented interior and exterior space, and converting these floor plans into watertight 3D meshes that are appropriate for simulations of building performance. We presented a validation of the effectiveness of the physical sketching environment for modeling and of our algorithm for automatically and correctly interpreting these designs. Response from both architecture and non architecture students about the system has been positive and encouraging.

Our current interpretation algorithm is quite successful at interpreting complex designs and produces reasonable results even for rather ambiguous sketches. We will continue to improve the algorithm, revising the rules for linking walls and defining separate interior spaces. We would like to incorporate domain-specific knowledge of common forms in architectural design and leverage symmetry within the sketch. Prior work in computer graphics demonstrates how approximate symmetries within a model allow decomposition, identification of correspondences, compression, warping to make the mesh more symmetric, and hole filling of missing data missing data [Golovinskiy et al. 2007; Pauly et al. 2008]. Furthermore, we plan to explore the automatic recognition of circulation paths within a design and generate appropriate roof overhangs and sloped roof shapes for the detected geometry.

We believe the core of the interpretation algorithm described in this paper can be extended to other forms of architectural sketching. For example, a direct digital equivalent of the physical environment with drag & drop, translation, and rotation of components would be straightforward. The system could also be adapted to a tablet display environment using existing sketch recognition technology for parsing straight and curved pen strokes.

References

AH-SOON, C., AND TOMBRE, K. 1997. Variations on the analysis of architectural drawings. In *In Proceedings of Fourth International Conference on Document Analysis and Recognition*, 347–351.

AH-SOON, C., AND TOMBRE, K. 2001. Architectural symbol recognition using a network of constraints. *Pattern Recognition Letters 22*, 2 (February), 231–248.

ALVARADO, C., AND DAVIS, R. 2001. Resolving ambiguities to create a natural computer-based sketching environment. In *In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann Publishers, 1365–1371.

AOKI, Y., SHIO, A., ARAI, H., AND ODAKA, K. 1996. A prototype system for interpreting hand-sketched floor plans. *Pattern Recognition, International Conference on 3*, 747.

AUTODESK, 2000-2008. Ecotect Analysis. `http://www.autodesk.com/ecotect-analysis`.

BEN-JOSEPH, E., ISHII, H., UNDERKOFFLER, J., PIPER, B., AND YEUNG, L. 2001. Urban simulation and the luminous planning table: Bridging the gap be-

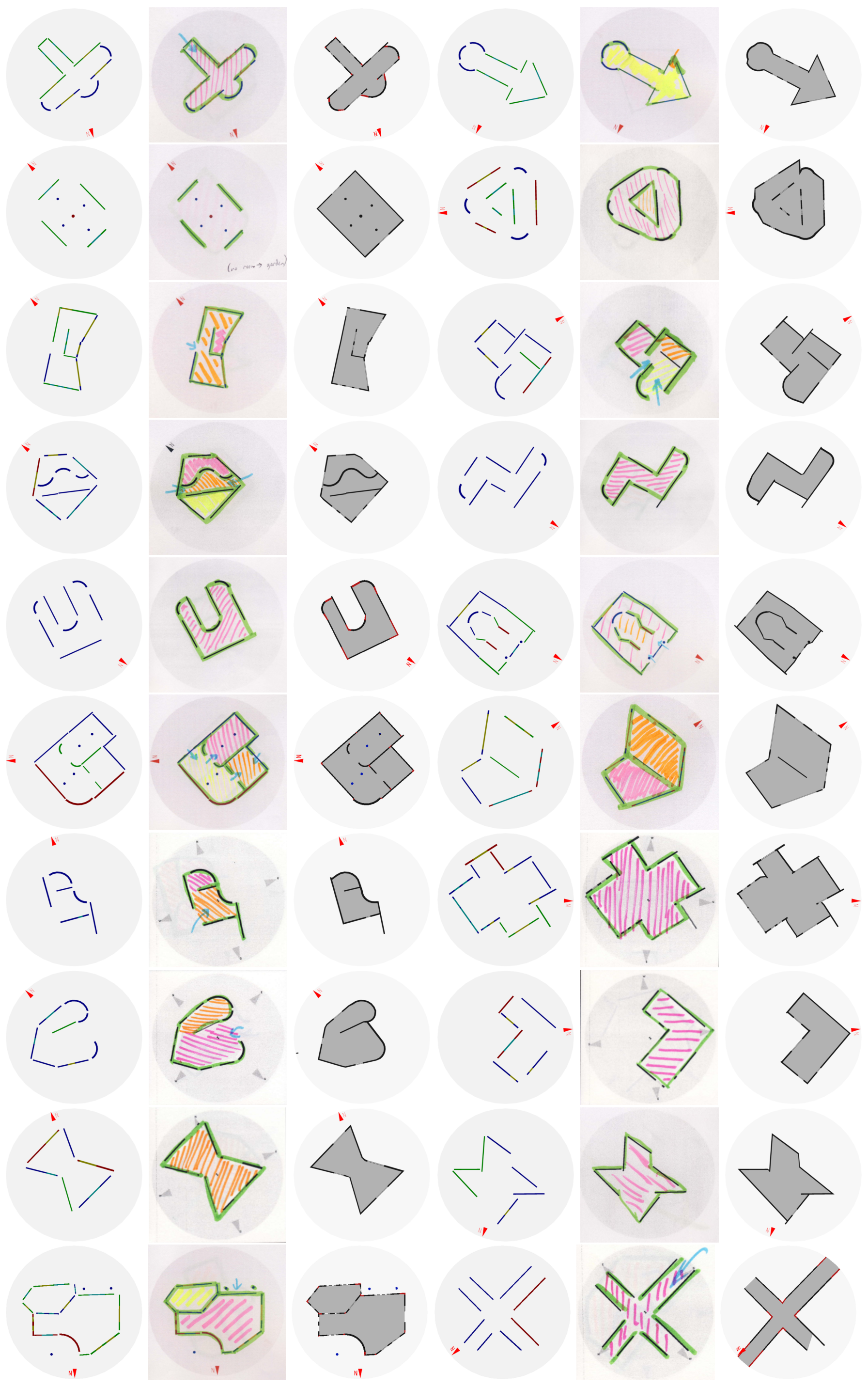

Figure 9: A sampling of the variety of collected physical sketch geometry: the original designer's annotation, and our automated interpretation of the interior/exterior space.

tween the digital and the tangible. *Journal of Planning Education and Research*, 195–202.

DESOLNEUX, A., MOISAN, L., AND MICHEL MOREL, J. 2002. Computational gestalts and perception thresholds. *Journal of Physiology - Paris 97*, 2003.

DORSEY, J., XU, S., SMEDRESMAN, G., RUSHMEIER, H., AND MCMILLAN, L. 2007. The mental canvas: A tool for conceptual architectural design and analysis. In *15th Pacific Conference on Computer Graphics and Applications*, 201–210.

GOLOVINSKIY, A., PODOLAK, J., AND FUNKHOUSER, T. 2007. Symmetry-aware mesh processing. *Princeton University TR-782-07* (Apr.).

GOOGLE, 2010. SketchUp: 3D modeling software. `http://www.sketchup.com`.

GROSS, M. D., AND LUEN DO, E. Y. 2000. Drawing on the back of an envelope: a framework for interacting with application programs by freehand drawing. *Computers & Graphics 24*, 835–849.

GROSS, M. D. 1994. Recognizing and interpreting diagrams in design. In *AVI '94: Proceedings of the workshop on Advanced visual interfaces*, ACM, New York, NY, USA, 88–94.

HAMMOND, T., AND DAVIS, R. 2007. Ladder, a sketching language for user interface developers. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, ACM, New York, NY, USA, 35.

IGARASHI, T., MATSUOKA, S., AND TANAKA, H. 1999. Teddy: A sketching interface for 3D freeform design. In *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, 409–416.

ITTI, L., KOCH, C., AND NIEBUR, E. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*, 11 (Nov), 1254–1259.

KANIZSA, G. 1979. *Organization in Vision: Essays on Gestalt Perception*. Praeger.

KOCH, C., AND ULLMAN, S. 1985. Shifts in selective visual attention: Towards the underlying neural circuitry. *Human neurobiology 4*, 4, 219–227.

KOFFKA, K. 1935. *Principles of Gestalt Psychology*.

KOUTAMANIS, A., AND MITOSSI, V. 1993. Computer vision in architectural design. *Design Studies 14*, 1, 40 – 57.

KOUTAMANIS, A. 1999. A framework for architectural sketch recognition. In *4th International Design Thinking Research Symposium on Design Representation*.

KULIKOV, V. 2004. *Building Model Generation Project: Generating a Model of the MIT Campus Terrain*. Master's thesis, Massachusetts Institute of Technology.

LANK, E., THORLEY, J. S., AND CHEN, S. J.-S. 2000. An interactive system for recognizing hand drawn UML diagrams. In *CASCON '00: Proceedings of the 2000 Conference of the Centre for Advanced Studies on Collaborative Research*, IBM Press, 7.

LAWSON, B. 2002. CAD and creativity: Does the computer really help? *Leonardo 35*, 327–331.

LIPSON, H., , LIPSON, H., AND SHPITALNI, M. 2002. Correlation-based reconstruction of a 3D object from a single freehand sketch. In *AAAI Spring Symposium on Sketch Understanding*, 99–104.

LLADOS, J., LOPEZ-KRAHE, J., AND MARTI, E. 1997. A system to understand hand-drawn floor plans using subgraph isomorphism and Hough transform. *Machine Vision and Applications 10* (August), 150–158.

LU, T., TAI, C.-L., SU, F., AND CAI, S. 2005. A new recognition model for electronic architectural drawings. *Computer-Aided Design 37*, 10, 1053 – 1069.

LUEN DO, E. Y. 2001. VR Sketchpad, create instant 3D worlds by sketching on a transparent window. In *Proceedings of CAAD Futures 2001 (Eindhoven*, Kluwer Academic Publishers, 161–172.

MACKIE, C., COWDEN, J., BOWMAN, D., AND THABET, W. 2004. Desktop and immersive tools for residential home design. In *Conference on Construction Applications of Virtual Reality*.

PAULSON, B. 2010. *Rethinking Pen Input Interaction: Enabling Freehand Sketching Through Improved Primitive Recognition*. PhD thesis, Texas A&M University.

PAULY, M., MITRA, N. J., WALLNER, J., POTTMANN, H., AND GUIBAS, L. 2008. Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics 27*, 3, #43, 1–11.

RAMAGUPTA, A., AND HAMMOND, T., 2007. Archiassist: A sketch recognition system for floor plans, March. Poster Presentation at Texas A&M Industrial Affiliates Program.

SHENG, Y., YAPO, T. C., YOUNG, C., AND CUTLER, B. 2009. A spatially augmented reality sketching interface for architectural daylighting design. *IEEE Transactions on Visualization and Computer Graphics*, 20 (November).

SHENG, Y., YAPO, T. C., YOUNG, C., AND CUTLER, B. 2009. Virtual heliodon: Spatially augmented reality for architectural daylighting design. In *Proceedings of IEEE Virtual Reality 2009*.

TOLBA, O., DORSEY, J., AND MCMILLAN, L. 2001. A projective drawing system. In *ACM Symposium on Interactive 3D Graphics*, 25–34.

WACOM, 2010. Cintiq: Interactive Pen Display. `http://www.wacom.com/cintiq/`.

ZELEZNIK, R. C., HERNDON, K. P., AND HUGHES, J. F. 1996. Sketch: An interface for sketching 3D scenes. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, 163–170.