EUROGRAPHICS 2016 / J. Jorge and M. Lin
(Guest Editors)

Volume 35 (2016), Number 2

SMARTCANVAS: Context-inferred Interpretation of Sketches for Preparatory Design Studies

Youyi Zheng[†1,3] Han Liu[†2,4] Julie Dorsey[3] Niloy J. Mitra[4,2]

ShanghaiTech University[1] KAUST[2] Yale University[3] University College London[4]

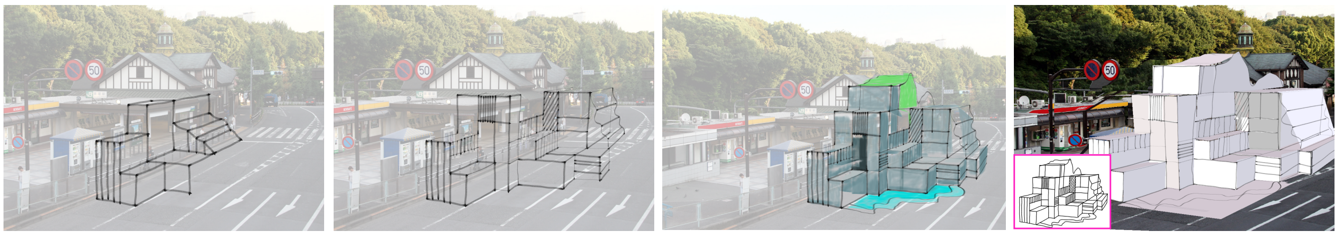


Figure 1: *Using a background photograph for anchoring, a practicing architect used our system to progressively develop a design. In the background, our algorithm lifts the sketched strokes along with their intersections to 3D using a novel context-based canvas arrangement extraction algorithm. The discovered canvases are painted and used for 3D-like interactions to facilitate preparatory design previews (right).*

Abstract
In early or preparatory design stages, an architect or designer sketches out rough ideas, not only about the object or structure being considered, but its relation to its spatial context. This is an iterative process, where the sketches are not only the primary means for testing and refining ideas, but also for communicating among a design team and to clients. Hence, sketching is the preferred media for artists and designers during the early stages of design, albeit with a major drawback: sketches are 2D and effects such as view perturbations or object movement are not supported, thereby inhibiting the design process. We present an interactive system that allows for the creation of a 3D abstraction of a designed space, built primarily by sketching in 2D within the context of an anchoring design or photograph. The system is progressive in the sense that the interpretations are refined as the user continues sketching. As a key technical enabler, we reformulate the sketch interpretation process as a selection optimization from a set of context-generated canvas planes in order to retrieve a regular arrangement of planes. We demonstrate our system (available at http:/geometry.cs.ucl.ac.uk/projects/2016/smartcanvas/) with a wide range of sketches and design studies.

1. Introduction

Good drawing, by virtue of intrinsic reciprocity between mind and act, goes beyond simple information, allowing one to fully participate in its significance, its life.

Michael Graves [Gra77]

Sketching is the essential tool for the early development of visual ideas, whether it is plans for a house, the layout of a plaza, a set design for the theater, or placement of a new bridge. Although sketching is guided by imagination, it is also referential when set in space, or when building on some previous idea or design. Naturally, when a sketch is made to record an observation, it is by definition anchored to a physical setting. Therefore, as the artist or designer draws and redraws, she not only refines the object, she also explores the physical setting in which the object or design will be placed. The renowned architect and theorist Michael Graves [Gra77] re-

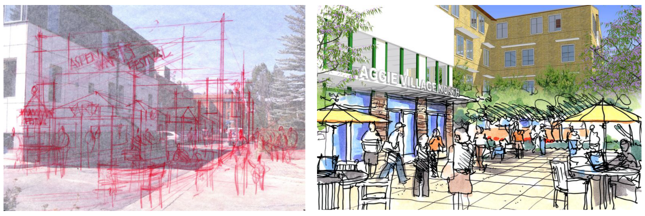


Figure 2: *Designers oversketch on photographs (left) or even SketchUp renderings (right) to anchor design ideas in the preparatory study stage of design.*

† Joint first authors.

ferred to this as a *preparatory study*, or variations on a theme, in which ideas are tested and questions explored.

One important technique designers use to set scale or to anchor ideas to existing features, is to oversketch on existing photographs, or even on crude SketchUp renderings. Oversketching is particularly helpful when planing refurbishments and informing design decisions. Figure 2 shows some examples, while we refer the readers to Jim Leggitt's *2010 Blog Collection* book for more examples.

But over-sketching only goes so far, as it is limited to a single viewpoint. The missing 3D information hinders exploration of the scene under different views or perspective perturbations, or preventing the user from selectively moving around objects, or conceptualizing larger design modifications. When the designer resketches to explore another view, unfortunately she cannot use or reference previous drawings.

Transforming such sketches to 3D is ill-posed. The sketched scenes are largely virtual and hence common vision-based reconstructions are not applicable; the sketches are rough, incomplete, and only indicative of the actual 3D curves, i.e., they lack any depth information; and often the sketches are from single viewpoints. If the designer chooses to design from the start in 3D using available or specialized CAD modeling tools, she is committed to a more restrictive, time-consuming and tedious approach that requires exact specifications and details from the start of the process. This invariably inhibits, rather than expands, the scope of exploration [JS11]. In effect, the designer trades some measure of artistic spontaneity and freedom for the precision of CAD software. For presenting early drafts of designs especially in architecture, it has been shown that sketches are preferred over CAD plots [SSLR96].

Interestingly, even crude 3D information can provide vital *context* to interpret 2D sketches. We investigate how rough 3D contextual information can be *directly* recovered simply based on a single background image or sketch. We observe that many architectural designs are of man-made scenes with primarily regular objects. Such objects are either mutually well-aligned, or near-regularly arranged. Hence, their embedding planes provide valuable *context* that can be leveraged to interpret oversketched lines. We present an interactive design abstraction tool for the artist and designer (see Figure 1).

To begin, the user calibrates the background image by annotating a few lines. As sketching continues, our system automatically partitions the sketched lines, groups them, and guesses their intersection pattern. When requested, our system *simultaneously* lifts all the sketched curves to 3D via a novel context-guided labeling formulation. Intuitively, by assuming the user strokes are planar, we can simultaneously assign embedding planes for all the strokes that lead to a regular arrangement in space. The process continues with the inferred abstraction providing enriched context for subsequent sketched strokes. Thus, as the scene context gets progressively enriched, sketch interpretation gets simpler. The extracted abstraction can then be used as billboard proxies for view perturbations and inspection for design previews (see supplementary video).

We tested our system on a variety of sketching and design scenarios in architecture, game design, interior/exterior planning, etc. We evaluated our system with multiple professionals receiving encouraging feedback. In summary, we: (i) characterize the problem of dynamic 2D sketch interpretation to produce 3D abstractions for preparatory design studies; (ii) solve the problem as a novel selection problem from a set of context-inferred candidate canvas planes; and (iii) develop an interactive tool realizing the algorithm, and test it on a variety of sketches and designs.

2. Related Work

Sketching in context. Artists and architects regularly sketch in the early, exploration phase (i.e., preparatory design). Usually a design takes shape within an existing spatial context by sketching atop a visual anchor, such as a photograph, a site plan, or even on Google SketchUp rigs of acquired data (cf., check streetscape improvements by Jim Leggitt, or other artists' works in the collection [JS11]). Previous design and analysis tools, such as 3D6B [Kal05], Mental Canvas [DXS*07], Sketching Reality [CKX*08a], and Insitu [PKM*11], were created to support such work flows, either by trying to fuse geometrically inconsistent strokes, or requiring multi-modal acquisition of the surroundings to act as scaffolds for subsequent design.

Sketch interpretation. Humans can fairly easily *infer* 3D forms from 2D sketches alone and researchers have long attempted to mimic the same algorithmically. Possibly the most famous system is the Teddy [IMT99], which lifts sketch lines to 3D as the user draw. Another approach is to use sketch lines for sketch-based retrieval of objects [EHA12, LS07, SC04], or co-retrieval of multiple objects to compose a scene [XCF*13]. The work of [CKX*08b] introduced a method to convert 2D sketches into 3D architecture model using a set of predefined primitives. Gingold et al. [GIZ09] convert user drawn 2D primitive shapes into 3D free-form surfaces by placing 3D primitives in agreeing with a set of user annotations which specify the geometric relations among the primitives such as length, angle, alignment, and symmetry. Similar idea was explicitly exploited in [LSMI10] where geometric annotations on 2D outlines are used to model man-made shapes on top of a photograph, or using arterial snakes [LLZM10] to abstract particular families of man-made objects.

The CrossShade system [SBSS12] lifts artistic cross-shade lines to 3D models by mathematically linking cross-sectional curves to model geometry, while Shao et al. [SLZ*13] proposed a system to interpret concept sketches to effective visualization of product designs with functional parts. More recently, the True2Form system [XCS*14] demonstrated how to mathematically model perception and design knowledge to create 3D forms from 2D sketches, while in a multiview context, [FLB15] capture dominant orientations of the real scene to anchor discovery of new orientations in the imaginary scene. For a detailed exposition, please refer to the survey book on line drawing interpretation [Coo08]. In contrast to these works, our system is to be used *from* the initiation of the design, rather than once the design has been completely sketched.

Sketch-based modeling. In the 3D modeling context, as an alternative to CAD-based modeling, various sketch-based metaphors have been proposed to give greater freedom to the users (see [OSSJ09] for a survey of earlier works). The work of iLoveSketch [BBS08] and its followup EverybodyLovesSketch [BBS09] introduced

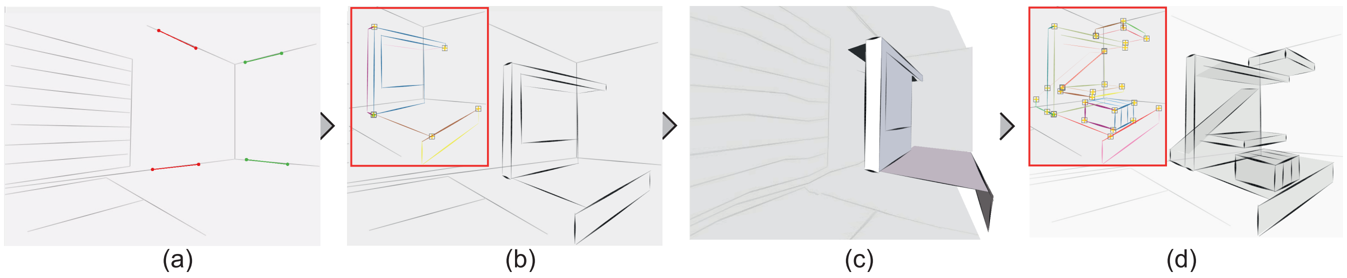


Figure 3: *Starting from a single image, the user annotates vanishing line information (a). She continues by sketching on top of the image. We estimate the respective vanishing information for each stroke with a set of automatically grouped strokes (different colors) exposed to the user (b). The user can modify the grouping results, or inter-stroke contact relations, then our system automatically lift a set of 3D canvases that host the strokes (c). Our system dynamically infers spatial structure from the user sketches (d).*

a sketch-based interface to construct curved 3D models by iteratively anchoring 3D canvases and projecting user strokes to the pre-anchored 3D planes. Schmidt et al. [SKSK09] propose the use of scaffold lines for analytical drawing. A marked difference is that we compute such planes automatically by analyzing the relations of user strokes to the given context. More recently, Saul et al. [SLMI11] introduced a specialized sketching system for creating chairs, Oztireli et al. [OUP*11] exploited symmetric curves to simplify 3D modeling, the Sketch2Design system [XXM*13] adapted and combined parts from 3D model collections based on user strokes for 3D modeling, while the PushPull++ [LWM14] introduced dynamic modeling tools to vastly simplify common modeling tasks. Shtof et al. [SAG*13] introduced a powerful constraint-based sketch modeling system, while exploiting inter-curve geo-semantic relations, while [SS14] proposed a flow-complex based shape reconstruction algorithm from a network of 3D curves. Our goal is also to facilitate simple, single-view sketch-based designing, but by exploiting the existing context to lift the sketches to 3D.

3. System Overview

The SMARTCANVAS system lifts user-drawn sketches to 3D by finding a set of embedding 3D planes, which we call *sweeping canvases*. By sweeping we refer to the notion that they are inherited by offsetting a set of reference canvas planes. First, using a standard procedure, we calibrate the background image (or sketch) using vanishing lines (see Figure 3(a)).

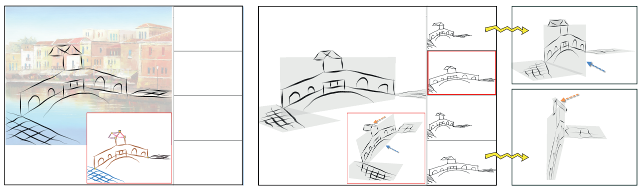


Figure 4: *Our system provides dynamic suggestions as the user continues to sketch. In the left figure, the user sketches are analyzed and grouped with annotations. Our system exposes intermediate canvases arrangements as suggestions and displays them to the user on a right UI panel (middle). The user can then choose the best arrangement which is in accordance with her sketches.*

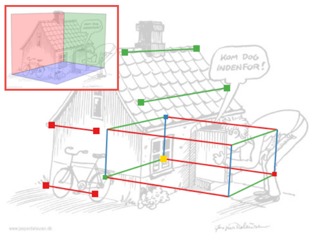
As the user sketches on an image, the strokes are dynamically classified into coplanar groups according to the reference vanishing directions. A coplanar group of strokes is called a *compound stroke*, or *c-stroke* for short. The *adjacency* relations of *c-strokes* are implicitly computed in our system, which is the key to our canvas optimization algorithm. The user can override the imprecise connections caused by severe occlusions, or annotate a set of new spatial relations, if desired. For example, she can assign orthogonal junction, hinge, or parallel relations between a pair of strokes. Note that all the user annotations, if any, are in 2D. The user does *not* have to interact in 3D for design sketching.

In the key algorithmic stage, we find a set of candidate canvases for each *c-stroke*, by sampling along the reference canvases and inducing from the relations to adjacent strokes if any. These candidate canvases are grouped by plane normals, forming a combination of canvas arrangements. We formulate the selection of canvas arrangements by solving a labeling problem, rank the solutions using corresponding optimization energies and display the top few suggestions (4 in our implementation) that best explain the inferred/annotated spatial relations. Each suggested arrangement is further refined using quadratic programming, and displayed on the right panel in the user interface (see Figure 4). User design is interpreted dynamically, i.e., the user can stop at any stage of sketching to view the 3D interpretation of the current strokes and continue sketching from any viewpoint. In Figure 4, the user stops sketching (left figure) and triggers the optimization by rotating the view (right figure). The top four candidate solutions are then listed on the right suggestion panel, with two of them zoomed in for showing the respective plane arrangements.

4. SMARTCANVAS Interface

In this section, we introduce the SMARTCANVAS interface to support user designs. Users can, at any time, intervene to override erroneous interpretations or define adjacency relations. The system consists of the following stages: camera calibration, dynamic sketching, stroke analysis and grouping, relation annotation, and canvas optimization.

Camera calibration. To calibrate the camera, we use a simple camera calibrating interface from PhotoMatch [Goo08]. Inset shows the user can adjust the two sets of parallel line segments (red and green) to indicate the x- and y-directions, so as to align with the respective vanishing directions. We assume that the focal point lies at the image center.



Once calibrated, we recover the camera matrix from the vanishing lines information [CRZ00], and create a 3D cuboid. The three orthogonal faces (upper-left subfigure) act as the initial reference canvases from which we sweep out candidate canvas planes.

Sketching. We use two modes for free-hand sketching. In a common mode, we allow free-hand sketches. In the user assisted mode, we smooth each stroke using cubic spline. If the user enables line fitting, we rectify a stroke to a line segment if the linear regression error is less than a threshold (2% of the stroke length). If the user draws a stroke which follows a previous drawn stroke, we override it with the new stroke. We used a simple ICP-based stroke matching mechanism (we consider two stroke to be coincident if there overlapping parts exceed a certain portion – 90% in our experiment). Users can easily switch between the two modes by pressing a keyboard shortcut.

In addition, during progressive sketching, the user can *copy and paste 3D sketches* generated by the system to avoid repetitive drawing. From user feedback, it not only saves the sketching time, but also presents a more accurate perspective. The system also provides perspective guidelines during sketching as in [BBS08].

Relation annotation. In our system, we are particularly interested in two types of relations between pairs of strokes. The user can approve, modify, or assign these relations:

(i) Contact. This relation refers to two stokes sharing a contact point such as the table-leg and the table-top. We show the user a set of highlighted cross-marks (see Figure 5), which we call, *connectors*, to indicate adjacent relations between stroke pairs. The adjacency is measured by overlapping 2D stroke points. Falsely detected connectors due to occlusion are easily overridden by pressing 'delete' or dragging the highlighted mark to reposition. To ensure that all strokes are connected (otherwise 3D recovery would be infeasible because of the DOFs), we highlight strokes without any contacts, i.e., isolated strokes to notify the user to edit.

Ambiguity is a key issue that affects 3D estimations of strokes. Hence, we provide a few tools to assist the user in avoiding ambiguities. First, the system only displays active connectors, i.e., once the user approves a 3D solution in a stage of sketching, the connectors of determined strokes will no longer visible unless the user choose so. Second, we simulate the function of layer. The user can select a few strokes marking as a layer, from where she continues sketching while unselected strokes fade out. The system then only detects contact relations among the strokes in the current layer.

(ii) Junctions and hinges. Our framework assumes that all user strokes are inter-connected and lie on some planar canvases. The canvases can be axis aligned, or induced from existing planes by intersection. We introduce two types of junction relations to support non-axis aligned canvases. In particular, a canvas A can form a junction with a reference canvas B, in which case A is orthogonal to B and passes through an *orthogonal junction* (see Figure 6a). The other case is the *hinge* relation, which means a stroke canvas form a hinge with another one (the ladder in Figure 6b). As shown in the figure, under junction mode, the selected connector is highlighted, so that the user holds the connector and drags it along a direction to assign the junction hinge (see also accompanying video).

5. Canvas Optimization

Our algorithm extracts coplanar strokes from free-hand user sketches. Let the set of coplanar strokes be $S = \{s_1, s_2, ..., s_n\}$ and the set of relations as $\Upsilon = \{\gamma_1, \gamma_2, \dots\}$. Our task is to find for each stroke s_i a best hosting canvas plane, p_i, such that the arrangement of the extracted canvas planes agree with Υ and the lifting of the 2D strokes via their corresponding canvas planes faithfully interpret the 3D structure of the 2D sketches. We design a multi-stage optimization by first finding a set of candidate planes for each stroke, and then progressively optimize the candidates to account for the annotated relations by quadratic programming, and finally select for each stroke the best embedding plane, and locally refine it. We denote the extracted plane that hosts a stroke as its *canvas*.

At a high level, our algorithm consists of a few key stages: auto-

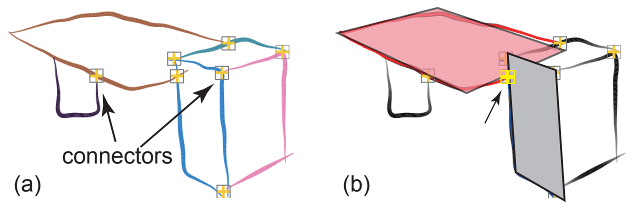


Figure 5: *Contact relations (yellow crosses) are represented as connectors in our system. Occlusions can lead to falsely detected contact relations. For example, in (a), the bottom-left connector between the black-brown strokes is detected at a false position, while the top-right connector between the blue-brown strokes is erroneously detected. (b) For editing contacts, we highlight the two corresponding c-strokes as the user hovers a* connector.

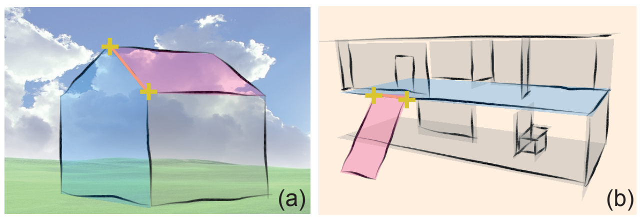


Figure 6: *We allow two types of junction annotations. The first type,* orthogonal junction *(a), indicates that one canvas (red) is orthogonal to an adjacent reference canvas (blue), and passes through the indicated line segment. The second type,* hinge relation *(b), indicates that one canvas (red) forms a rotational angle with a reference canvas (blue), and passes through the indicated line.*

matically coupling the input strokes, generating a set of candidate reference canvases, assigning the strokes to appropriate canvases using a global probability-based assignment, and a final canvas refinement step to locally adjust the canvas positions. We perform the above steps dynamically in the background, and the reference canvases get augmented as soon the user locks to any suggestions. We now explain the steps in details.

We classify canvases into two main sets and adaptively treat them in the subsequent optimization: one set of canvases G_1 are those that are parallel to some initial reference canvases, the other set of canvas G_2 that form junction or hinge relations to the canvases in G_1 (see also Figure 6). We now detail the steps.

Grouping coplanar strokes. User drawn strokes, if found to be coplanar, are automatically grouped into a compound stroke, *c-stroke*. When the user draws a new stroke s_j, a stroke will correspond to one or multiple vanishing directions (see details below). We analyze the new stroke with all its adjacent strokes detected in screen space and search for the largest group $g := \{s_j, s_k, \dots\}$ that satisfies the following condition: the union of their corresponding vanishing points contains no more than two vanishing directions, i.e., they can be hosted by a common plane. If found, we group all the strokes in group g.

We continuously perform this analysis in the background. Once the user finishes, we expose the auto-grouping results and the user can override any false groupings. Specifically, she can select a few strokes and regroup them. A false grouping typically occurs when there are occlusions or the existence of curved profiles where the line decomposition fails. Figure 7 shows a complete grouping process for a simple desk model. Once we group all strokes, we estimate their candidate canvases.

Initial reference canvases. By default, the set of initial canvases contains the three orthogonal face orientations from the cuboid that was created at the camera calibration stage, denoted as $I = \{r_1, r_2, r_3\}$, labeled as red, green, and blue planes in Figure 8. Note that as the optimization progresses, new candidate directions and canvases get added to the reference set. Further, the user can specify additional planes as reference canvases (e.g., by specifying a directional line on top of the image), which are used for generating candidate canvases for non-axis aligned objects.

Stroke candidate canvases. Each stroke s_i in G_1, i.e., those strokes that are not being specified to form a junction or hinge re-

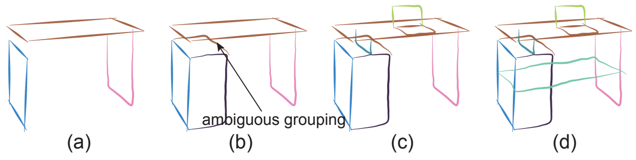


Figure 7: *Our auto-coupling algorithm progressively collects and groups the user strokes (a-d). Once a new stroke is drawn, we detect its vanishing directions, and check its contacts with the existing strokes. We find the most confident coupling group by estimating coplanarity, indicated by different colors. The user can override the automatic suggestion.*

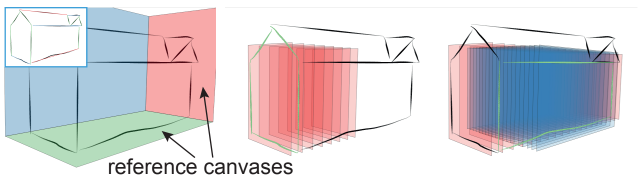


Figure 8: *We sample candidate canvases for a compound stroke (shown in green) by sweeping across the existing reference canvases. In this example, the stroke contains multiple vanishing directions, and hence multiple sets of candidate planes are sampled. We only show a few of them for visual clarity.*

lation to other strokes, by assumption, should be hosted by a canvas that is parallel to one of the existing reference canvas $r_j \in I$. We then create the set of candidate canvases C_i for s_i by sweeping $K = 30$ planes along the $\pm$ normal direction of r_j (see Figure 8). To ensure the validity of the candidate canvases, we discard any swept canvas that does not contain any stroke point in screen space or the length of projected stroke (in 3D) is more than the 10× the perimeter of the canvas.

Blindly enumerating all the possible canvases in C_i, however, is expensive. Further, it can lead to degenerate cases (e.g., all strokes lie on the same plane). Hence, as an important observation, we find that most of the strokes are planar and can be broken into sub-strokes that are aligned with the vanishing directions, or alternatively parallel to the edges of canvases in I. This observation allows us to filter out those obviously unsuitable canvases.

For each stroke in G_1, our goal is to find the canvas in I that best matches the stroke profile. Specifically, for each user stroke s_i, we recursively split the strokes into two sub-strokes by the method of Ramer-Douglas-Peucker (RDP). For each splitting, we measure how well the sub-strokes agree with the vanishing lines compared to its original shape. If the average score increases, we stop the splitting. We measure the score as a directional angle distance from the line segment to a vanishing point. Specifically, given a vanishing point $\mathbf{v}$, and a line segment $\mathbf{pq}$, we define their distance as:

$$d(\mathbf{pq}, \mathbf{v}) := \left| \frac{\mathbf{v} - (\mathbf{p}+\mathbf{q})/2}{\|\mathbf{v} - (\mathbf{p}+\mathbf{q})/2\|} \cdot \frac{(\mathbf{q}-\mathbf{p})}{\|\mathbf{q}-\mathbf{p}\|} \right|. \tag{1}$$

Given the distance function, their angle distance is defined as $\theta(\mathbf{pq}, \mathbf{v}) = \arccos(d(\mathbf{pq}, \mathbf{v}))$. We fit a *line segment* for each sub-stroke, and measure its angle distance to each of the three vanishing points $V = \{\mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_z\}$. Note that we do not break any stroke that is approximately a straight line by checking whether the maximum distance from stroke points to the fitting line is less than the RDP threshold.

Stroke-canvas probability assignment. We now look for candidate canvases for each c-stroke. For a c-stroke s, let us denote its sub-strokes as $s = \{s^1, s^2, \dots, s^n\}$. For each s^k, we compute its probability of being coincident with a vanishing direction $v_t, t \in \{x, y, z\}$, as

$$P_t(s^k \rightarrow \mathbf{v}_t) = \begin{cases} 1.0 - \frac{\theta(\mathbf{pq}^k, \mathbf{v_t})}{\pi/2} & \text{if } \theta(\mathbf{pq}^k, \mathbf{v_t}) \le \pi/4 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

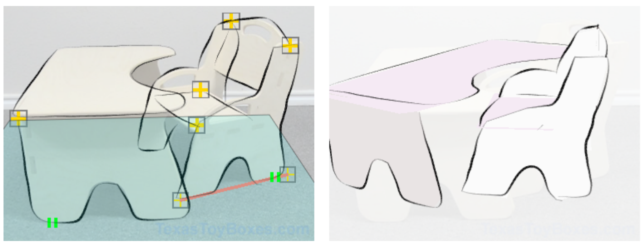


Figure 9: *In order to model a non-axis aligned object, we allow users to assign a reference line segment (shown in salmon in the left figure). The reference line assigns new candidate canvases by shifting the existing reference canvases. Together with the axis-aligned objects, we recover the underlying canvases for both objects.*

Here, $\mathbf{pq}^k$ is the line segment that approximates sub-stroke s^k. We do not normalize the probability, as the probability confidence is used to select the possible normal directions of a c-stroke. Recall that the normal of each initial reference canvas corresponds to one vanishing direction $\mathbf{v}_t \in V$. Thus, for c-stroke s, we compute the probability confidence of its normal direction being coincident with a vanishing direction v_t as:

$$P(s \to \mathbf{v}_t) = \sum_{(\gamma_1,\gamma_2,...,\gamma_n)\in\Theta} \prod_{\substack{k=1,\\ \gamma_k \neq t}}^{n} P_t(s^k \to \mathbf{v}_{\gamma_k}). \tag{3}$$

Here, Θ is all possible enumeration of $(\gamma_1,\gamma_2,...,\gamma_n)$ where $\gamma_k \in \{x,y,z\}/\{t\}$. We sort $P(s \to \mathbf{v}_t)$-s for all $\mathbf{v}_t, t \in \{x,y,z\}$. Our task now is to determine which $\mathbf{v}_t$-s are candidate reference directions and select the corresponding initial reference canvases whose normal are coincident with $\mathbf{v}_t$-s. There can be up to three normal directions for each c-stroke, especially if a stroke is not coincident with any vanishing direction, in which case all the three vanishing directions will be the candidate reference directions.

Let us denote the sorted list of directions as $\Delta = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ with decreasing $P_i(s)$. We search through Δ and look for a $\mathbf{v}_i$ whose probability confidence $P(s \to \mathbf{v}_i)$ merits a significant drop compared to previous mean value, i.e., $P(s \to \mathbf{v}_i)/(\frac{1}{i}\sum_{j=0}^{i-1} P(s \to \mathbf{v}_j)) < \tau$. Once we find such a value, we discard all $\mathbf{v}_{j,j \geq i} \in \Delta$. τ is s user specified value which we set as a fixed value of 1/2.

Canvas selection. For each c-stroke s_i, we now have a set of candidate canvases $C_i = \{c_i^1, c_i^2, \ldots\}$. Our goal is to select from this set the best one $s_i \to c_i^k$ such that the spatial relations among the selected canvases are consistent with the spatial relations Υ. Note that the spatial relations are non-local, the selection involves assigning canvases to all the strokes simultaneously, rather than sequentially.

Essentially, we have a labeling problem: For each stroke s_i, our goal is to select only *one* of its candidate canvases, i.e., select a label l_i from the set of all its candidate canvases labeled as $\{l_i^0, l_i^1, \ldots\}$. The selected canvases should be in accordance with Υ. We define unary and binary terms to score the selections.

The unary term is to impose a hard constraint that a stroke selects a canvas only from its own label set, and not from the candidate label set of another stroke. Thus, $E(s_i \to l_j^k)$ is given equal weight (set to 1) if $i = j$, and high penalty (set to ∞) for $i \neq j$, for any k.

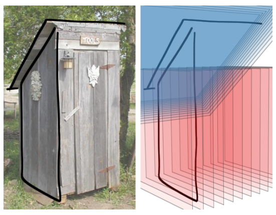
The binary assignment likelihood term is defined as: $E(s_i \to l_i^k, s_j \to l_j^l) := \exp(\phi(s_i^k, s_j^l)^2)$ where, s_i^k denotes the embedded stroke of s_i to canvas $c_i^k \in C_i$ corresponding to the label l_i^k and similarly for s_j^l. The function $\phi(s_i^k, s_j^l)$ is estimated as the minimum distance between the two embedded strokes in 3D if the strokes s_i and s_j share an adjacent relation in R, otherwise it is given a small weight (set to 0.1). Here, the minimum distance between two strokes refers to the minimum distance among any pair of 3D stroke points in s_i and s_j respectively.

Finally, we can extract the best labeling as:

$$\{l_i\}^\star := \underset{\{l_i\}}{\operatorname{argmin}} \sum E(s_i \to l_j^k) + \sum_{i,j} E(s_i \to l_i^k, s_j \to l_j^l). \tag{4}$$

We used alpha-expansion algorithm [KZ04] to solve this.

Handling junction and non-axis aligned strokes. The set of strokes in G_1 that participate in the labeling process are those whose canvases are parallel to some reference canvas r_i. We now include the set of strokes in G_2 that shares *orthogonal junction* relations to the strokes in G_1 in the labeling.



In order to sample the candidate planes for the type of orthogonal junction strokes, we derive from the existing sampled canvases in G_1. Specifically, if s_j (in G_2) forms an orthogonal junction relation with $s_i \in G_1$, we generate one candidate canvas c_j^l from each of the candidate canvas c_i^l of s_i such that $c_j^l \perp c_i^l$ and c_j^l passes through the projection of user annotated junction line in c_i^l. The inset figure shows the candidate canvases sampled for the roof stroke which forms an orthogonal junction relation with the left wall.

For the strokes that are annotated with hinge relations to other stroke, we leave them out in the initial canvas selection, and revisit them in a next stage of optimization. We offer a simple user interface to let the user specify a reference junction line to indicate its rotated direction around some reference canvas that is parallel to the ground (we assume the non-axis aligned object has upright orientation). We then recover the x- and y-vanishing points, sample candidate canvases for each of the stroke in the object in the same way as above, and solve the labeling formulation using MRF (Markov Random Field) formulation as described before. Figure 9 shows an example.

Canvas refinement. The selected best candidate canvas for each stroke, however, was sampled in a discrete space. Thus, the contact relations among the spatial canvases might remain loose (see Figure 10).

For a stroke s_i, denote its selected canvas as $c_i = (\mathbf{o}_i, \mathbf{n}_i)$ from the previous step, where $\mathbf{o}_i$ and $\mathbf{n}_i$ are the plane center and normal respectively. We look for a best offset $\lambda_i : \mathbf{o}_i' = \mathbf{o}_i + \lambda_i \mathbf{n}_i$, such that the embedded strokes are in close contacts with each other. If a canvas c_k shares an orthogonal junction relation with another canvas, the parameter to optimize is still an offset λ_k as the canvas orientation remains. If a canvas c_k is attached as a hinge to some other canvas, we optimize for the best rotational angle θ_k for s_k (Note that the position of the canvas is dependent on the other canvas). Thus, we minimize the objective function:

$$f(\lambda_1,\lambda_2,\ldots,\theta_k,\ldots) = \sum_{(i,j)\in\Upsilon} [\phi(s'_i - s'_j)^2 + \gamma(s'_l,\vartheta)^2], \\ \text{s.t.} \quad \varsigma_i < \lambda_i < \tau_i, \quad -\pi < \theta_k < \pi \qquad (5)$$

where, s'_i and s'_j are the embedded strokes in $\mathbb{R}^3$; $\gamma(s'_l,\vartheta)$ is a distance function to ensure a projected stroke s'_l touches the ground plane ϑ if assigned; ς_i and τ_i are the minimum and maximum value for λ_i to keep the canvas visible in screen space when sliding along the normal direction; and finally, $\varsigma_i = \tau_i = 0$ if the canvas was already optimized in a previous stage. We solve this quadratic programming using the Levenberg-Marquardt algorithm. As an example, Figure 10c and 10d show the results of MRF selection and canvas optimization, respectively.

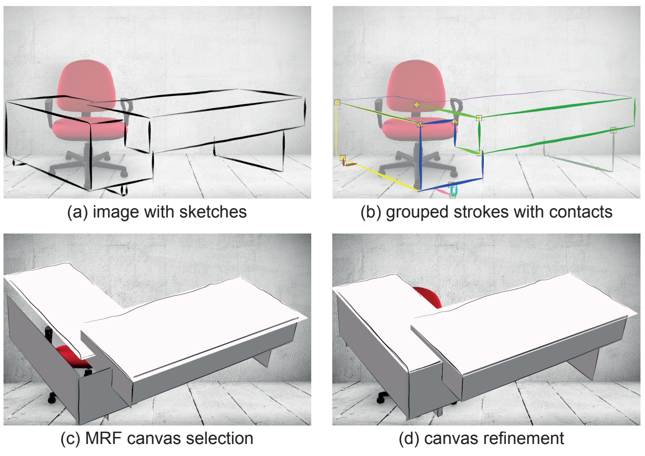


Figure 10: *We lift the 2D sketch lines (a), (b) to 3D via extracted embedding canvas planes as a selection problem finding the best canvas arrangement that agrees with Υ (c). The selected canvases are then refined by a quadratic programming that minimizes the contacts distance while complying with user annotated constraints (d).*

Dynamic interpretation. We observe that in a typical design process, a user often tends to stop at intermediate steps, check the current results, refine it or continue to sketch. Such a stop-and-examine approach enables the designer sketch from different viewpoints, and surprisingly benefits the system in reducing the risk of accumulating errors during incremental sketching.

We design a dynamic sketch interpretation mechanism to monitor the interpretation ambiguities on-the-fly. An ambiguity aries if (i) user strokes severely deviates from the given perspective of the background image; or (ii) occlusions leading to failed automatic detection. We check if any group of strokes merits more than one possible interpretation, i.e., there exists close probability confidences $P_i(s)$ for groups of canvases with different normals. In this case, we perform the above optimization for each such group and in the meanwhile pop up potentially possible spatial arrangements of canvases on a right panel. The user can check to see if the current arrangement in the primary viewer is correct and choose to override the current arrangement with a better one on the right panel if any (Figure 4).

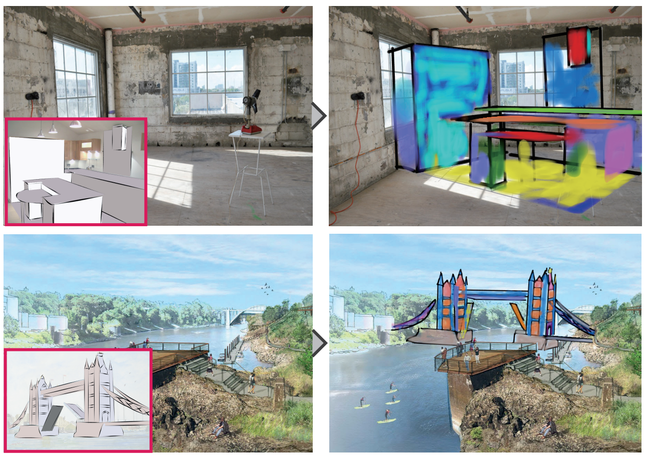


Figure 11: *Transfer of design contexts. The user can easily create 3D abstractions from existing images, as well as rework and transfer them into new contexts. Both the source- and target-scene configurations in 3D are rectified seamlessly. We used an airbrush interface for 3D painting on the extracted canvases.*

For dynamic suggestion, we consider the current c-stroke which has ambiguities, divide its candidate canvases into groups of $\{g_1, g_2, ...,\}$ such that each group of canvases have the same normal directions. We fix one group g_i, and perform the above optimization, to find the best canvas arrangement a_i. By enumerating all possible groups, we get multiple candidate arrangements of canvases. For each arrangement, we measure its score by the following term:

$$S_{a_i} = [\frac{1}{|S|}\sum_{s_i\in S} P(s_i \rightarrow \mathbf{v}_{\mathbf{s_i}}|a_i)] \times f(\lambda_1,\lambda_2,\ldots,\theta_k,\ldots|a_i). \qquad (6)$$

Here, $\mathbf{v}_{s_i}$ is the direction of the canvas of s_i and $f(\ldots|a_i)$ is the cost function in Equation 5, given the arrangement a_i. We rank all the a_i-s by their scores and list the top four candidate arrangements on the right panel, or all arrangements if the number of candidates is less than 4. As the number of ambiguous groups at incremental sketching stages are typically few, and hence enumerating is not expensive.

6. Results and Evaluation

We used our system to sketch a variety of designs anchored to man-made environments. These included furniture, interior designs, architecture, and illustrations from children's books.Figures 11, 13 and 15 show various results (see also supplementary video).

Presenting sketch abstractions. The output of our system is a set of 3D canvases that embed and abstract the user's strokes. Figure 12 shows a design sketch of an unfurnished kitchen using the SMARTCANVAS system and the finished kitchen for comparison. We present the algorithm output using shaded canvases overlaid with 3D line drawings (Figure 12(a)). For this, we adopt the rendering style of Cole et al. [CGL*08]. For cleaner visualization,

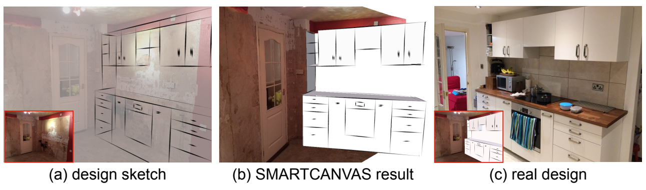


Figure 12: *We used SMARTCANVAS to design of a kitchen by sketching on top of a photograph of the unfurnished kitchen (a). SMARTCANVAS was used to generate previews to make design decisions (b). Figure (c) shows the finish kitchen.*

we additionally crop any canvas based on the hosted stroke profile(Figure 12(b)), if a compound stroke forms a nearly closed curve. For complicated elements, such as trees, people, etc., the user manually crops the embedding plane. To mimic traditional drawing media and provide a more appealing look, we stylized the strokes.

In our discussions with architects and designers, they overwhelmingly favored the sketchy, abstracted versions of the presentation over a more complete rendered versions using full 3D geometry. They noted that they prefer 'the sketchy, incomplete' look, as it encourages and invites critique and reinterpretation of designs, rather than indicating a definitive, final design.

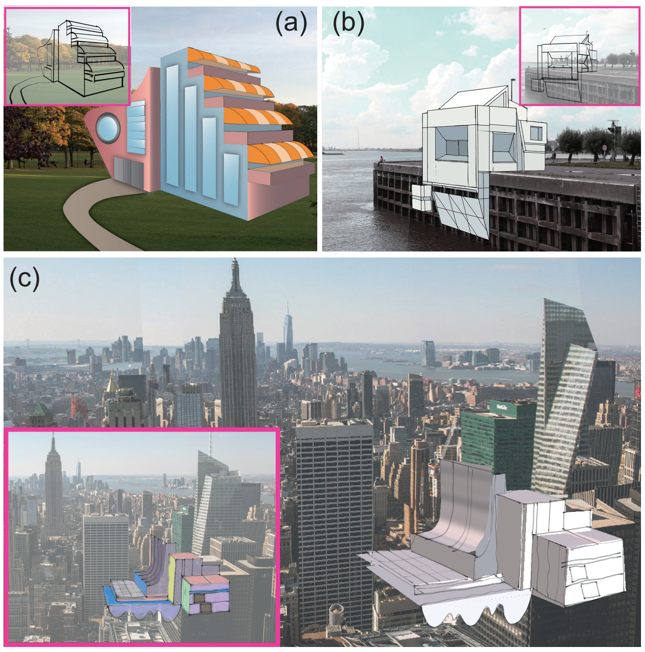


Figure 13: *Different concepts developed by a designer (a) and a practicing architect (b, c) using our system.*

Design reflections. In the most critical evaluation of our system, we worked with three well-known practitioners to get feedback on our system. The first user is an architect who is also an accomplished digital artist and illustrator. He frequently directly sketches on top of photographs as Photoshop-layers as part of his workflow.

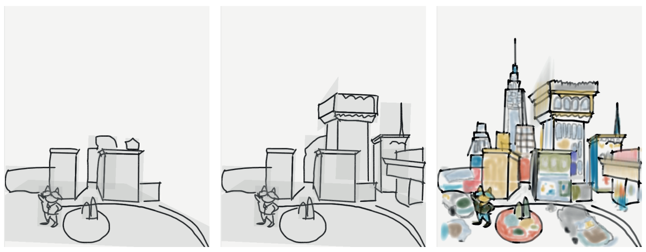


Figure 14: *Line drawing of a city view is adaptively anchored in 3D. The original drawings are from a children's book illustrator.*

For 13(b), he used such stroke layers as input to our system, and then investigated the generated abstractions to preview the scene. He later developed Figure 1 and 13(c) directly in our system, without any prior work. Please refer to the video for a playback of the full session for Figure 1. His feedback was very positive and he was excited to be able to get this quick feedback without having to spend, as he put it, "hours on a CAD system." He did comment on having to learn the user interface for accepting/changing contacts among curves, but he noted that in the end this is just learning "a convention system." He felt that the tool will be particularly useful in "discussing designs with clients." More fundamentally, he noted that the tool collapses the sketching and 3D modeling stages into one, which is attractive as "not everyone in our firm can model in 3D."

Our next user is a designer. His sketching style primarily employed planar curves on paper, see Figure 13(a), that we scanned into the drawing system. He appreciated that he could continue to simply sketch, though he was initially worried that he would need to draw in correct perspective. We demonstrated that he could use the system to alter the viewpoint.

Finally, our third user is a children's book illustrator, who frequently creates interactive scenes (see Figure 14). She used our system to get previews of her sketches, perform view perturbations, and further develop her drawings. She appreciated the ability to quickly "navigate in 3D" as she was drawing and to experience the scene in a more spatial way than paper would allow. She noted, "After working in this system, it will be hard to go back to 2D."

Note that the SMARTCANVAS interface was updated twice based on initial feedback from the first user (that is, the architect). For larger scale evaluation, the project code/demo will be made publicly available.

User study. We also had a about 10 casual volunteers experiment with the system. Figure 15 shows some examples created by this user group. Each of them took about 2-3 hours to get familiar with the system and then complete their first sketch, since several of our test users had never used a 3D system previously – or, in one case, a digital drawing system of any kind. More experienced digital artists picked up the system in about 30 minutes. The hardest part for designers was getting accustomed to the process of providing corrective annotations. A majority of them, many of whom have a computer graphics/vision background, preferred to calibrate the camera themselves so that they could have a better idea of the perspective during sketching. They liked the guidelines of vanish-

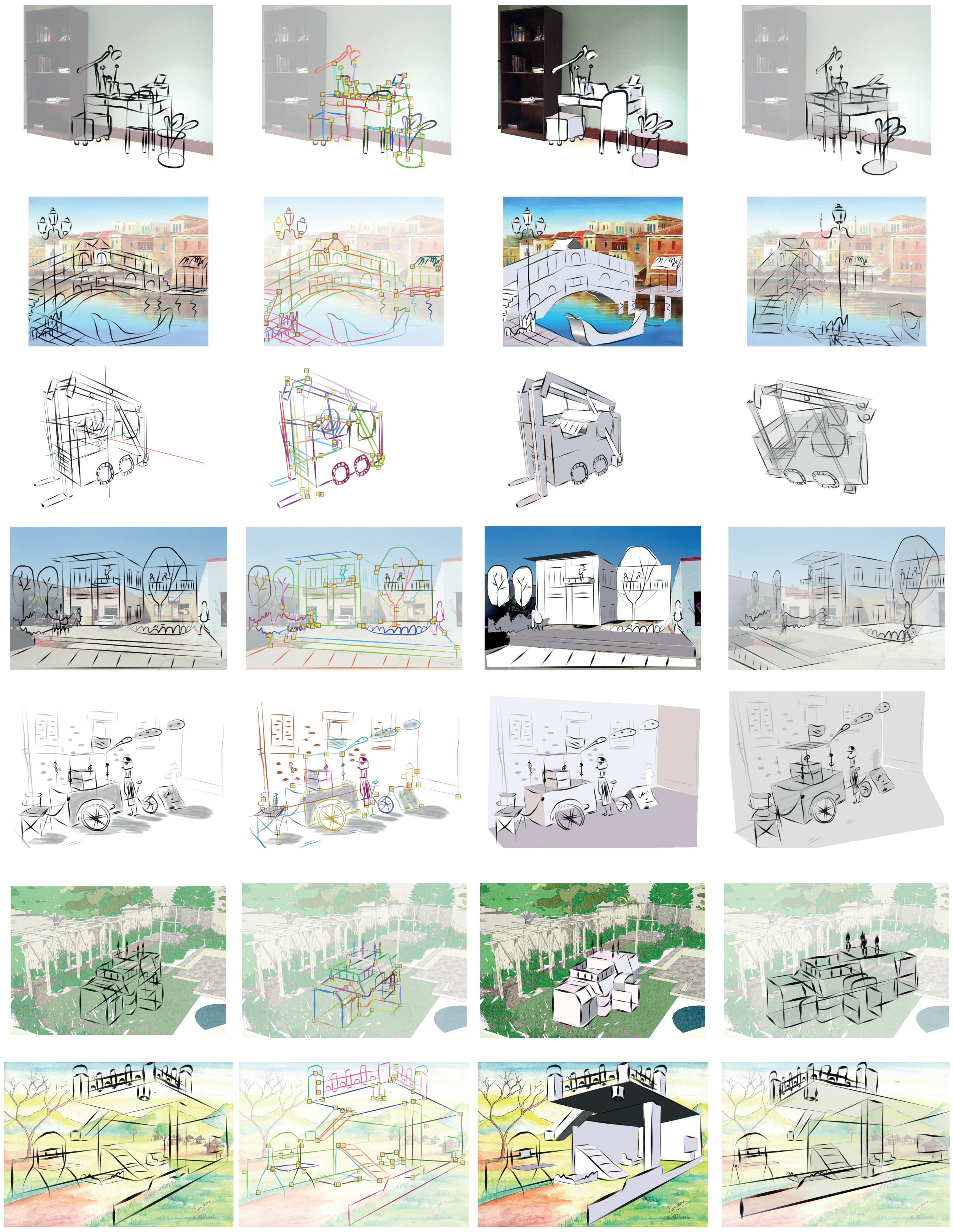


Figure 15: *Context-anchored design examples developed using our system.*

ing directions and the aids for correcting perspective, as sketching with an accurate perspective is otherwise difficult for non-artists. In all of these cases, the background image was used to generate a coordinate system to anchor the sketch.

Statistics. Table 1 presents the number of strokes, connectors, user annotations, and the average modeling time for the various examples shown in the paper. Note that the majority of the time was spent sketching. We tried a version of the system using digital-pen input, i.e., Wacom Bamboo tablet. However, the interface was still indirect (the user needs to touch on a separate pad to get correct mouse position on the screen). Experimenting with various input devices is an area for future work.

In the simple examples, the automatic grouping and connector suggestions were sufficient. In other cases, the user had to edit a few grouping or connector suggestions (e.g., the first example in Figure 15). Such editing or erasing operations, however, are fairly easy to perform since the false detections occur mostly at occlusion and improper stroke positions. It typically took users less than a minute to investigate and correct such grouping and contacts. User intervention can be avoided completely for those who prefer to sketch from different views. For example, in Figure 13(c), the architect did not use any assisted tools as he frequently rotated the camera to avoid ambiguity. As shown in the Table 1, the number of stroke groups is half of the number of strokes, and there are 21 stroke groups having just one single stroke. In contrast, there are fewer stroke groups in Figure 16-top but with more group corrections. The average manual intervention in grouping is less than 2% among all users, and the average relation annotation, i.e., hinge and junction, is less than 1 across all the results. Subsequently, our selection and optimization algorithm runs at interactive rate. The MRF selection takes longer, about 2-3 seconds per 100 strokes. The quadratic programming is less than a second thanks to the Levenberg-Marquardt algorithm and a good initialization from the selection stage. All experiments were done on a laptop with an Intel 2.7GHz CPU and 4GB memory.

Quantitative evaluation. Our framework localizes 2D sketches into rough 3D, whereas the 3D positions of the inferred canvases are derived from images. Hence, it would be interesting to see to what extent the inferred 3D space is in accordance with the 3D structure of the original image. As ground truth, we first downloaded models from the Trimble 3D Warehouse, took their renderings as screenshots, and oversketched to follow the image features. To evaluate the recovered geometry, we compared the accuracy of the generated 3D abstraction from our system with ground-truth 3D models to measure the quality of the canvas planes.

Figure 16 shows two abstracted scenes with the extracted 3D canvas planes being visualized (in cyan) against the original model (in orange). Interestingly, our algorithm working just on the sketch lines can still recover the original geometry fairly accurately. The average point-to-point error of the two reconstructed models are 0.0232 and 0.019 respectively, after normalized in a unit box. Noticeable errors occur at regions where the sketches are not covered, e.g., the ground and some of the occluded faces.

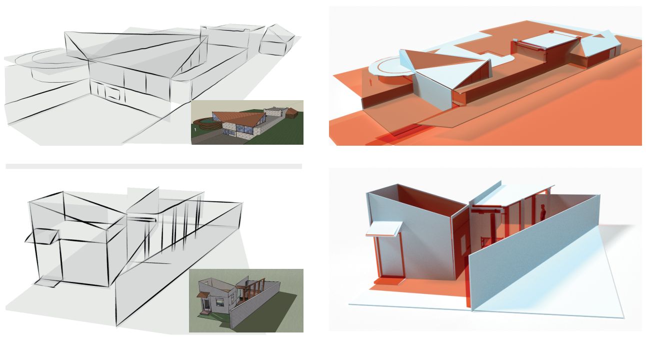
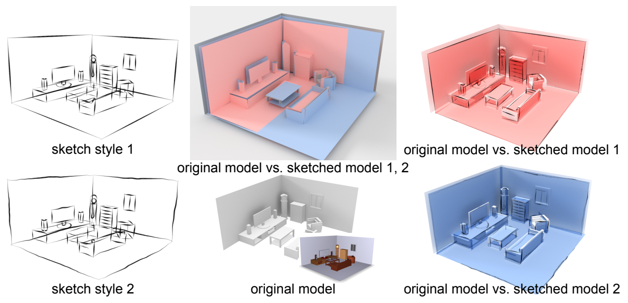


Figure 16: *Evaluation of our algorithm against 3D ground truth. Inset images in the left column are sketches contouring the screenshot of 3D models. The right column shows our reconstructed results (cyan) overlaid with original model (shown in orange).*


sketch style 1
original model vs. sketched model 1, 2
original model vs. sketched model 1
sketch style 2
original model
original model vs. sketched model 2
Figure 17: *Our algorithm is tolerant to variations in user sketches. Style 1 (a) uses straight lines, while style 2 (b) uses freehand strokes. The two output canvas abstractions are visually and structurally similar compared to each other, as well as the original model (middle and right subfigures).*

Sensitivity to sketching variations. We tested the robustness of our algorithm with different styles of sketching. Figure 17 shows the abstractions for a sketched living room model from two varying input sketches compared to the ground-truth model: one uses nearly straight lines, which are aligned with image perspective, while the other has more freehand strokes. Our algorithm is not very sensitive to such variations. Visually, the outputs are dependent on the position of the contact points along the strokes, as they impose constraints on the spatial locations of the final canvases. Sketching closer to the image perspective results in more precise interpretation of the original image/sketch objects. In our experience, the main source of error is incorrect connectors.

Limitations. Our system is primarily designed to handle man-made structures and scenes, where regular arrangements of planes (canvas) exists, and can be extracted. Hence, the system is not suitable for abstracting freeform buildings and structures. That said, however, as shown in a few of the examples, the system does handle planar curves and can contain (ruled) surfaces spanned by two such parallel planar curves. Further, in the camera calibration step, we do assume a single camera view. However, many hand-sketched examples (e.g.,Figure 14) have multi-perspectives spread across the scenes. While our system can still extract a reasonable camera for

Table 1: *Statistics of our method. Time includes time for sketching, grouping, annotation and optimization.*

Figure	# strokes	# connectors	# connector correction	# annotation	# grouping correction	# stroke groups	time (m)
1	155	241	0	0	0	76	36
3	50	34	3	1	0	16	5
13 (c)	71	91	0	0	0	36	10
16 (top)	79	49	5	4	5	19	10
15 (top)	92	67	8	2	6	33	20

one of the multiple views, the camera leads to noticeable distortions in other parts of the scene (under rotation).

7. Conclusion

We presented an interactive system to dynamically interpret 2D sketches of man-made environments by abstracting the sketched strokes using arrangements of canvas planes. The extracted canvases effectively lift the 2D sketches to 3D, and thus allow novel preview possibilities to artists at design time to assist in adapting and developing ideas. On the technical side, we presented a labeling approach to achieve this goal by utilizing the set of context-generated candidate planes to interpret scene sketches. We presented evaluation results of our system on a range of designs, sketches, and usage scenarios.

Considerable effort in graphics and vision research has tried to resolve sketches into well-defined shapes. SMARTCANVAS takes a different approach by operating at a middle ground that seeks to selectively use context from imagery to provide structure aimed at supporting and informing sketching. Clearly more work remains to be done to understand sketching across many creative fields, especially coupled with structure-aware shape synthesis [LVW*15], and to understand and define the role computer graphics can play.

Acknowledgements

We thank the anonymous reviewers for their invaluable comments, suggestions, and additional references. We would like to thank Amal Aboulhassan, Pooya Sareh, Wenwen Zhu, Lubing Fan, Cristina Amati, and Dongming Yan for their great help on the user study. We specially thank Luke Pearson (http://www.alephograph.com/) for contributing his sketches using the system. The project was supported in part by the ERC Starting Grant SmartGeometry (StG-2013-335373) and the US National Science Foundation Award (No. 1302267).

References

[BBS08] BAE S.-H., BALAKRISHNAN R., SINGH K.: Ilovesketch: As-natural-as-possible sketching system for creating 3d curve models. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (2008), UIST '08, pp. 151–160. 2, 4

[BBS09] BAE S.-H., BALAKRISHNAN R., SINGH K.: Everybodylovessketch: 3d sketching for a broader audience. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2009), UIST '09, ACM, pp. 59–68. 2

[CGL*08] COLE F., GOLOVINSKIY A., LIMPAECHER A., BARROS H. S., FINKELSTEIN A., FUNKHOUSER T., RUSINKIEWICZ S.: Where do people draw lines? *ACM Trans. Graph. 27*, 3 (Aug. 2008), 88:1–88:11. 7

[CKX*08a] CHEN X., KANG S. B., XU Y.-Q., DORSEY J., SHUM H.-Y.: Sketching reality: Realistic interpretation of architectural designs. *ACM TOG 27*, 2 (Apr. 2008), 11:1–11:15. 2

[CKX*08b] CHEN X., KANG S. B., XU Y.-Q., DORSEY J., SHUM H.-Y.: Sketching reality: Realistic interpretation of architectural designs. *ACM Trans. Graph. 27*, 2 (May 2008), 11:1–11:15. 2

[Coo08] COOPER M.: *Line Drawing Interpretation*. Springer-Verlag London, 2008. 2

[CRZ00] CRIMINISI A., REID I., ZISSERMAN A.: Single view metrology. *Int. J. Comput. Vision 40*, 2 (2000), 123–148. 4

[DXS*07] DORSEY J., XU S., SMEDRESMAN G., RUSHMEIER H., McMILLAN L.: The mental canvas: A tool for conceptual architectural design and analysis. In *The 15th Pacific Conference on Computer Graphics and Applications* (2007). 2

[EHA12] EITZ M., HAYS J., ALEXA M.: How do humans sketch objects? *ACM TOG (SIGGRAPH) 31*, 4 (2012), 44:1–44:10. 2

[FLB15] FAVREAU J.-D., LAFARGE F., BOUSSEAU A.: Line drawing interpretation in a multi-view context. In *CVPR* (2015). 2

[GIZ09] GINGOLD Y., IGARASHI T., ZORIN D.: Structured annotations for 2D-to-3D modeling. *ACM Transactions on Graphics (TOG) 28*, 5 (2009), 148. 2

[Goo08] GOOGLE, INC.: Google Sketchup. 2008. 3

[Gra77] GRAVES M.: The necessity of drawing: Tangible speculation. *Architectural Design* (June 1977). 1

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: A sketching interface for 3d freeform design. In *SIGGRAPH '99* (1999), pp. 409–416. 2

[JS11] JONES W., SAGOO N.: *Architects' Sketchbooks*. Thames and Hudson, 2011. 2

[Kal05] KALLIO K.: 3D6B Editor: Projective 3D Sketching with Line-Based Rendering. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2005), Jorge J. A. P., Igarashi T., (Eds.). 2

[KZ04] KOLMOGOROV V., ZABIH R.: What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 2 (2004), 147–159. 6

[LLZM10] LI G., LIU L., ZHENG H., MITRA N. J.: Analysis, reconstruction and manipulation using arterial snakes. *ACM TOG (SIGGRAPH Asia) 29*, 6 (2010), 152:1–152:10. 2

[LS07] LIPSON H., SHPITALNI M.: Optimization-based reconstruction of a 3d object from a single freehand line drawing. In *ACM SIGGRAPH 2007 courses* (2007), ACM, p. 45. 2

[LSMI10] LAU M., SAUL G., MITANI J., IGARASHI T.: Modeling-in-context: User design of complementary objects with a single photo. In *Proc. SBIM* (2010), pp. 17–24. 2

[LVW*15] LIU H., VIMONT U., WAND M., CANI M.-P., HAHMANN S., ROHMER D., MITRA N. J.: Replaceable substructures for efficient part-based modeling. *CGF (EUROGRAPHICS)* (2015). 11

[LWM14] LIPP M., WONKA P., MÜLLER P.: Pushpull++. *ACM Trans. Graph. 33*, 4 (July 2014), 130:1–130:9. 3

[OSSJ09] OLSEN L., SAMAVATI F. F., SOUSA M. C., JORGE J. A.: Technical section: Sketch-based modeling: A survey. *Comput. Graph. 33*, 1 (2009), 85–103. 2

[OUP*11] Öztireli A. C., Uyumaz U., Popa T., Sheffer A., Gross M.: 3d modeling with a symmetric sketch. EG. 3

[PKM*11] Paczkowski P., Kim M. H., Morvan Y., Dorsey J., Rushmeier H., O'Sullivan C.: Insitu: Sketching architectural designs in context. *ACM TOG (SIGGRAPH Asia) 30*, 6 (2011), 182:1–10. 2

[SAG*13] Shtof A., Agathos A., Gingold Y., Shamir A., Cohen-Or D.: Geosemantic snapping for sketch-based modeling. *Computer Graphics Forum 32*, 2 (2013), 245–253. Proceedings of Eurographics 2013. 3

[SBSS12] Shao C., Bousseau A., Sheffer A., Singh K.: Crossshade: Shading concept sketches using cross-section curves. *ACM Transactions on Graphics (SIGGRAPH Conference Proceedings) 31*, 4 (2012). 2

[SC04] Shesh A., Chen B.: Smartpaper: An interactive and user friendly sketching system. In *Computer Graphics Forum* (2004), vol. 23, Wiley Online Library, pp. 301–310. 2

[SKSK09] Schmidt R., Khan A., Singh K., Kurtenbach G.: Analytic drawing of 3d scaffolds. In *ACM TOG (SIGGRAPH Asia)* (2009), vol. 28, p. 149. 3

[SLMI11] Saul G., Lau M., Mitani J., Igarashi T.: Sketchchair: An all-in-one chair design system for end users. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction* (2011), TEI '11. 3

[SLZ*13] Shao T., Li W., Zhou K., Xu W., Guo B., Mitra N. J.: Interpreting concept sketches. *ACM TOG (SIGGRAPH) 32*, 4 (2013). 2

[SS14] Sadri B., Singh K.: Flow-complex-based shape reconstruction from 3d curves. *ACM Trans. Graph. 33*, 2 (Apr. 2014), 20:1–20:15. 3

[SSLR96] Schumann J., Strothotte T., Laser S., Raab A.: Assessing the effect of non-photorealistic rendered images in cad. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (1996), ACM, pp. 35–41. 2

[XCF*13] Xu K., Chen K., Fu H., Sun W.-L., Hu S.-M.: Sketch2scene: Sketch-based co-retrieval and co-placement of 3d models. *ACM Transactions on Graphics 32*, 4 (2013), 123:1–123:15. 2

[XCS*14] Xu B., Chang W., Sheffer A., Bousseau A., McCrae J., Singh K.: True2form: 3d curve networks from 2d sketches via selective regularization. *ACM TOG (SIGGRAPH) 33*, 4 (2014). 2

[XXM*13] Xie X., Xu K., Mitra N. J., Cohen-Or D., Gong W., Su Q., Chen B.: Sketch-to-design: Context-based part assembly. *Comput. Graph. Forum 32*, 8 (2013), 233–245. 3