Magic Canvas: Interactive Design of a 3-D Scene Prototype from Freehand Sketches

HyoJong Shin Takeo Igarashi

The University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
113-0033 Tokyo, Japan
shin@ui.is.s.u-tokyo.ac.jp takeo@acm.org

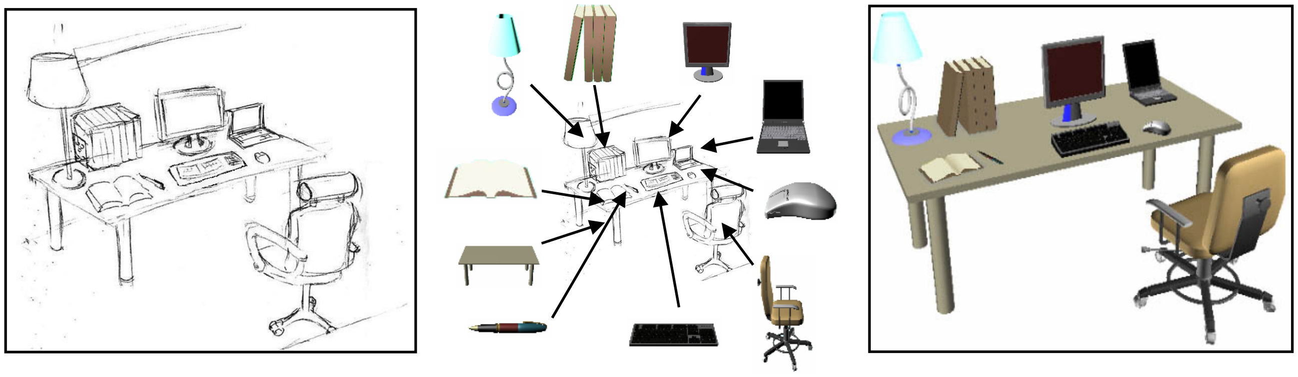

Figure 1. An example of 3-D scene construction from a sketch in our system. The system transforms the initial 2-D sketch (left) to a 3-D scene (right) by retrieving and placing corresponding models in a database (center). The process is interactive; the user performs sketch and selects an appropriate model and posture with the aid of the system.

Abstract

Construction of a 3-D scene consisting of multiple objects can be tedious work. Existing 3-D editing tools require the user to choose an appropriate model in a database first and then carefully place it in the scene at a desired position combining various operations such as translation, rotation, and scaling. To simplify the process, we propose a system that takes simple 2D sketches of models in a scene as input for 3D scene construction. The system then automatically identifies corresponding models in a database and puts them in the appropriate location and posture so that their appearance matches the user's input sketches. The system combines a 3-D model search and a 3-D posture estimation to obtain the result. This system allows the user to construct a prototype of a 3-D scene quickly and intuitively.

We conducted a user study to compare our interface with traditional menu-based UI and verified that our system was useful for constructing a 3-D scene prototype, especially for facilitating the exploration of various alternative designs. We expect our system to be useful as a prototyping tool for 3-D scene construction in various application areas such as interior design, communication, education, and entertainment.

CR Categories: I.3.6 [Computer Graphics]: Methodology and Techniques – Interaction Techniques

Keywords: prototype, sketch, 3-D scene

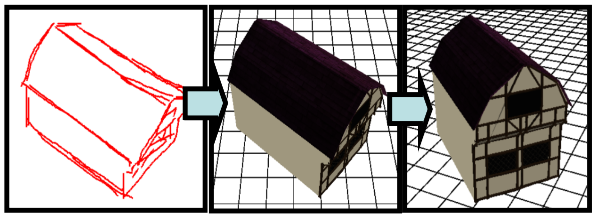

Figure 2. The basic operation of the Magic Canvas system. The user sketches a 2-D model (left); the system then searches for an appropriate 3-D model and arranges it to fit to the sketch (center). The model from different angles (right).

1 Introduction

Three-dimensional (3-D) scenes are applied in various fields, such as video games, interior design, animation, and feature films. The design of 3-D scenes usually starts from concept sketches drawn by a designer off-line, and then 3-D modelers create a corresponding 3-D scene prototype. However, it is tedious and difficult to convert a 2-D sketch into a 3-D scene that includes multiple objects using existing graphics tools because the user must repeat a series of operations, such as translation, rotation, and scaling, iteratively.

To facilitate construction of a 3-D scene prototype, we propose a sketch-based interface, Magic Canvas, for designing 3-D scenes consisting of multiple models in a database. The user can create a desired scene by just drawing a 2-D sketch depicting its appearance. The user first draws rough sketches of desired models in the scene and the system searches for the most appropriate models from a database. The system then adjusts the position and posture of the models so that their rendered images on the screen match to the user's sketches. It frees the user from manual import of a suitable model and manual placement of the model using editing commands in existing 3-D tools. This system has potential application in areas other than prototyping tools for 3-D scene design. For example, it can serve as entertainment for nonprofessionals and as educational software for students learning the concepts of perspective projection.

We briefly discuss related work on 3-D imaging and then describe details of the system's user interface and its implementation. We further report the results of a user study we ran to examine usability of the system. The user study showed that this system was useful for replacing models and changing postures for comparison with other models. Although the results indicated that the current implementation performs well compared to a standard interface, feedback from participants revealed a need for fine-tuning of the interfaces.

2 Related Work

2.1 Sketch-based 3-D Modeling

Many studies have proposed methods for constructing a 3-D model from user-defined 2-D drawings. These include the reconstruction of rectilinear models covered by planar faces achieved via constraints solving [1] or using optimization-based algorithms [11,12], and the reconstruction of the 3-D geometry of a 3-D curve using energy minimization [9] or symmetric relationships [13]. Our specific interest is in interactive sketching interfaces for designing 3-D models using 2-D gestures. The SKETCH system [14] is used to design 3-D scenes consisting of simple primitives, and the Teddy system is used to design free-form models [6]. Several extensions of the original Teddy system have been proposed [8]. Transformation strokes system [15] allows users to assemble existing 3D models and make a new model or a scene by means of single strokes. Our system is most closely related to the SKETCH system, but we combine database searches to support the construction of 3-D scenes consisting of existing 3-D models.

The above systems use sketching as a tool to construct traditional 3D models, but some systems explore the possibility of using sketching itself as a new design medium. The projective stroke system [18] projects the user's strokes onto a sphere surrounding the viewpoint and the Harold system [19] projects strokes to billboards to represent a quasi-3D scene. Bourguignon et al.'s system allows the user to place strokes in the air that indicate local surface contours [20]. We take a similar approach for representing the initial sketch but also provide a way to transform this sketch into a complete 3D model.

2.2 Sketch-based Retrieval

Funkhouser et al. [4] proposed a sketch-based retrieval system known as the 3-D Model Search Engine. The types of query that the user can input include a 3-D model made by Teddy [6] and 2-D sketches from three different views. We extend their work and add functionality to place the model in the scene at an appropriate position, orientation, and scale. Fonseca et al. [3] proposed a sketch-based retrieval system that searched clip-art items from a database, but this system limited its retrieval target to 2-D vector drawings.

2.3 Finding Camera Parameters

Although 3-D objects are usually rendered through a camera that is already established in the 3-D world, we sometimes need to estimate camera parameters from a projected 2-D image. Various camera calibration methods [2] have been widely used in the field of computer vision for determining camera parameters in 3-D space based on a 2-D photograph. Gleicher et al. [5] proposed a novel camera control method where the camera is controlled by constraints on the screen.

In this paper, we propose a method to estimate the posture of a 3-D model based on a user's sketch. Whereas the basic principles are the same, we slightly modified typical camera parameter estimation algorithms to deal with the high ambiguity (imprecision) of the input sketch.

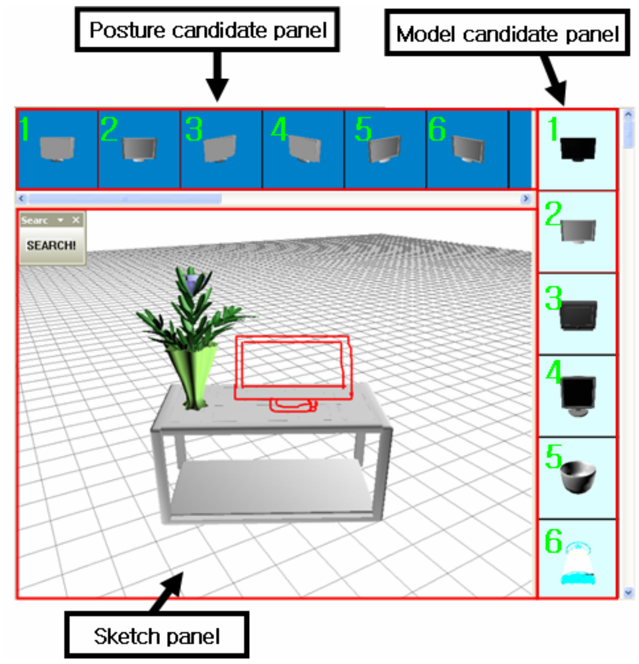

Figure 3. Screen shot of the Magic Canvas system. The numbers in the model candidate panel and the model posture panel represent retrieval rankings.

3 User Interface

Our system works as a 3-D scene construction system for scenes consisting of prefabricated 3-D models in a database. The user interactively draws the desired appearance of a model on the screen and the system places the 3-D model in the appropriate position, orientation, and scale. We also provide a candidate selection interface [17] to deal with the inherent ambiguity in hand-drawn sketches. In addition to showing the model with the highest matching score in the 3-D scene in the most probable posture, the system presents other possible candidates in the database as well as other possible postures so that the user can quickly consider alternatives.

The current system uses a database consisting of 200 ~ 300 3-D models. The user can construct the database in many ways. For example, the user can download models from the internet or make a pool of models provided by model designers if he or she is in a design-related company.

3.1 Screen

This system is composed of three panels: a sketch panel, a model candidate panel, and a model posture panel (Figure 3). The sketch panel is where the system displays the 3-D scene and the user sketches. The model candidate panel shows candidates of 3-D objects from the retrieval results, and the model posture panel presents posture candidates of the selected model.

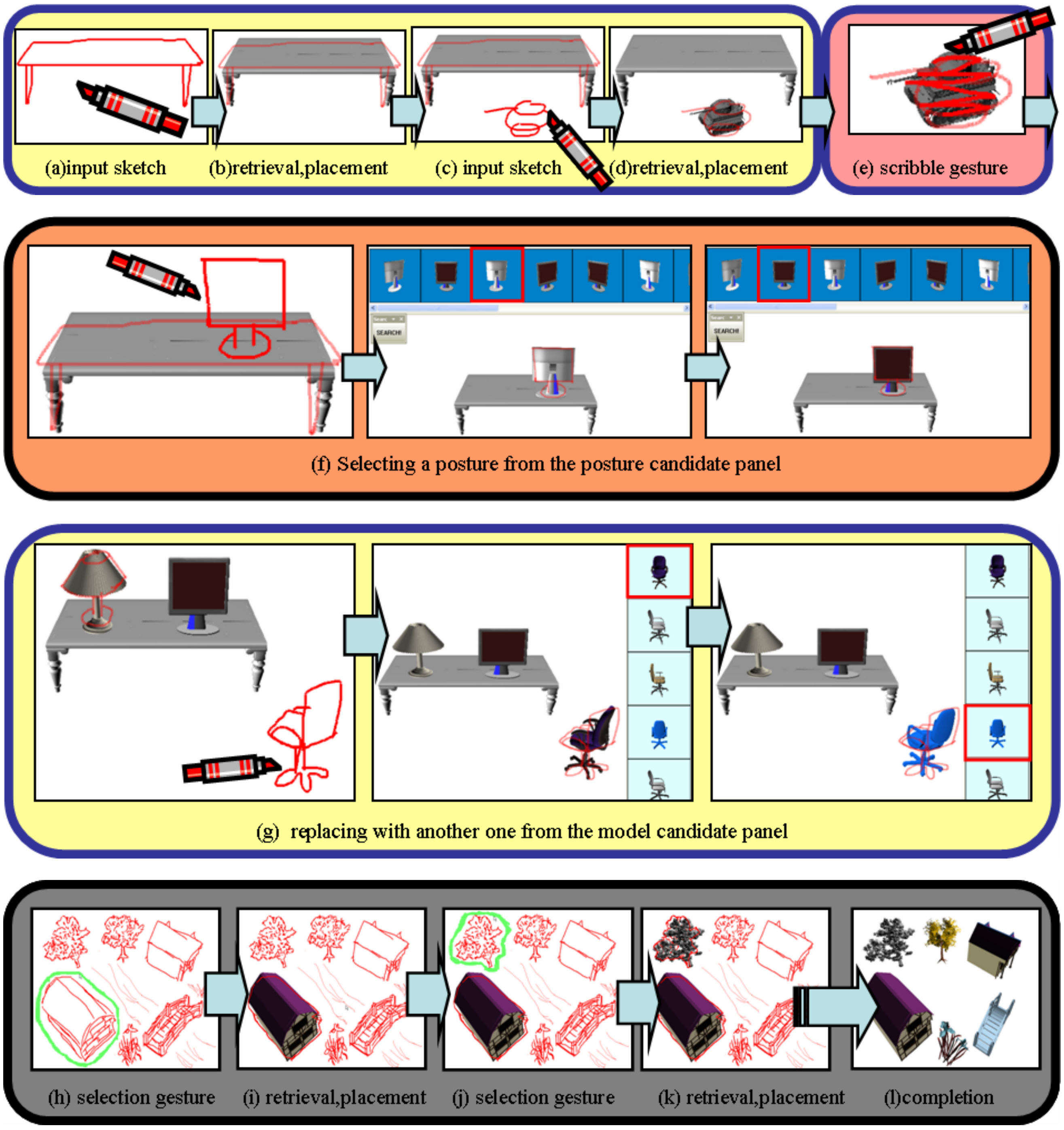

Figure 4. Interface overview.

3.2 General Procedure

The user draws a rough sketch in the sketch panel (Figure 4a, c). We assign left-mouse-dragging to a free-form line drawing and right-mouse-dragging to camera control. After finishing the sketch of a model, the user pushes the search button and the system shows the candidates in the model panel in the order of matching scores. The model that has the highest retrieval score appears in the sketch panel and is fit to the sketch automatically (Figure 4b, d). The user can draw scribble gesture on the target strokes or the model to delete them (Figure 4e). This system allows the user to choose other postures in the posture panel when he or she wants to change the posture (Figure 4f).

The user can replace the model with another candidate model by clicking it in the model candidate panel (Figure 4g). The user can also begin scene construction from a bitmap image. The user interactively selects a set of strokes in the bitmap image that represents a model using lasso selection and the system places a corresponding 3-D model in the scene. (Figure 4h–l).

3.3 Arrangement and Deletion of Models

The user can place a 3-D model by sketching the projected appearance of the model. Magic Canvas allows the user to draw any number of strokes in an arbitrary order. It also lets the user add an arbitrary number of ornament-scribbles inside the outline. Furthermore, it is possible to put a new model on top of an already arranged model by drawing strokes on it. This "stacking" method is borrowed from the SKETCH system [14].

Model adjustment is tedious work using existing tools because they must manually adjust the position of the model combining translation, scaling, and rotation. This process is especially laborious when using perspective projection because it is difficult to recognize the exact size of the model. The user often brings a model in the scene, adjusts its size at the initial location, moves the model to the desired position, and then notices that the size is inappropriate due to the perspective effect. In our system, the user can intuitively specify the proper size of the model by directly drawing the desired image. If a model is not satisfactory, the user can delete it by scribbling on the target model.

3.4 Replacement of Models

The designer sometimes wants to compare different versions of a 3-D scene by replacing one model with a similar model. For example, suppose that many bed models with similar shapes exist in the database (Figure 5). If the user is employing existing tools, he or she would have to reiterate the same work performed in placing the current model to switch to a new model. The more models are available, the more variety of scenes the user may want to try. As a result, the process can be a very tedious and time-consuming.

With Magic Canvas, if the user wants to replace a specific model that is already arranged by the user, he or she just needs to click the model on the floor. The system searches the database again and shows the candidate models on the model panel because the system remembers the original sketch information drawn by the user. The user can replace the selected model by clicking another candidate model shown in the model candidate panel. Thus, the user can easily explore many different versions of the scene without repeating tedious operations.

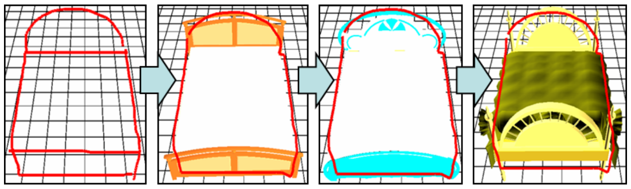

Figure 5. A bed model is replaced by an alternative model.

3.5 Using Bitmap Images

Some users might want to use existing graphics software they are familiar with like Photoshop or Illustrator to create a 2-D sketch. Some users might want to sketch directly onto physical paper when they are not accustomed to drawing on a computer.

To satisfy those needs, our system allows the user to import a bitmap image drawn by a designer. After importing an image into the system, the user converts individual models in the scene by selecting corresponding parts of the sketch by lasso selection. The system takes the sketch inside the lasso as a query to the search system, as is the case with online sketching. The limitation of current system is not supporting recognition of an image in which multiple objects are overlapped.

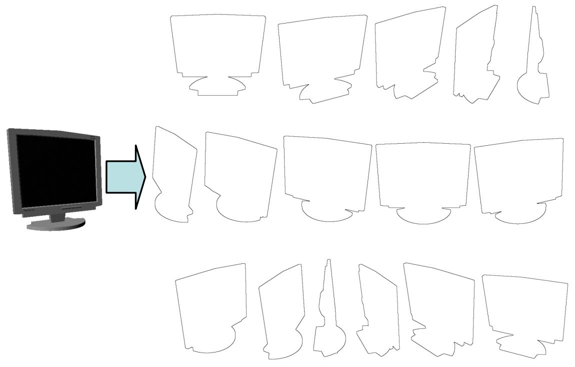

Figure 6. Sixteen contours of a computer monitor model from a quarter-view.

4 ALGORITHMS

The system consists of two major components. One is model retrieval from the database and the other is positioning of the model in the scene. We describe each of these components separately in this section.

4.1 Retrieval from the Database

Models in the database are indexed using the feature vectors generated by examining the contours of the model rendered from 16 reference views (Figure 6). We use quarter-views to generate these reference views because people tend to use a quarter-view when they are asked to draw a 3-D scene consisting of multiple models. This is in contrast to the observation that people tend to use a front view or side view when they are asked to draw a single model [4]. When the user completes a sketch of a model and presses the search button, the system generates corresponding feature vectors from the sketch and returns the models in the database that have the most similar feature vectors.

Our system uses two types of feature vectors, a global feature and a local feature, to improve retrieval performance. For a global feature, we use the Centroid Fourier Descriptor [10]. Steps for making a feature vector are summarized as follows. First, the system traces the outermost contour of the model as in Figure 7(b). Second, the system measures the distance from the center to the outermost part of the contour or a sketch(Figure 7c) and plots it in angular space (Figure 7d). Finally, the data are transformed to frequency domain data by applying a discrete Fourier transform (Figure 7(e)).

The actual sampling process is a bit more elaborate. Uniform angular sampling around the center results in uneven sampling around the boundary when the aspect ratio of the sketch is very large or small. When the sketch is stretched horizontally, the top and bottom regions are oversampled and the side regions are undersampled (Figure 8a). To obtain a more uniform distribution, the system decides sampling direction by dividing the four edges of a bounding box into equal numbers of segments (Figure 8b). The system then casts a ray from the center to the sampled points on the bounding box and identifies the farthest intersection with the given sketch (or the rendered image of a model in the database).

The Fourier transformation generates the global feature vector we use. We calculate the difference between two feature vectors as follows:

$$FV = \sum_{i=0}^{n} \left| Freq_{sketch,i} - Freq_{model,i} \right|,$$

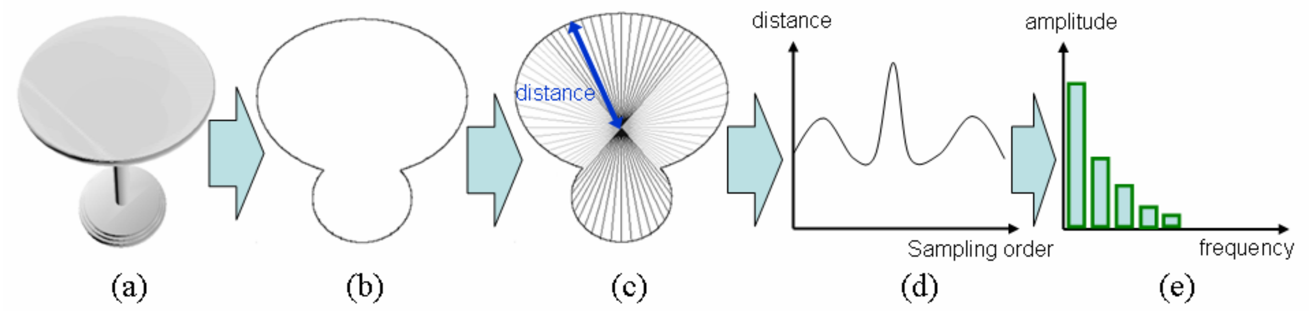

Figure 7. Examples of (a) the original model, (b) the outermost contour of (a), (c) taking sampling points, (d) radii data, (e) frequency data.

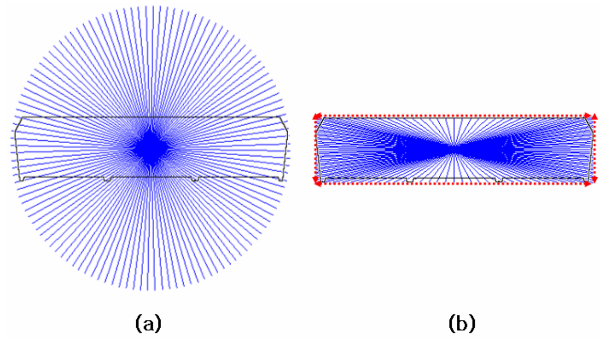

Figure 8. How to make a sampling point. (a) the top and bottom regions are oversampled and the side regions are undersampled in uniform angular sampling (b) the top, bottom and side regions are sampled uniformly in bounding box sampling.

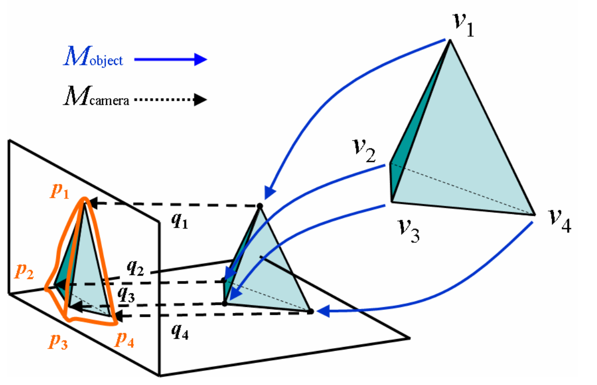

Figure 9. Adjusting posture. The system calculates the matrix **M***object* to fit the 3-D model to the orange user's sketch.

where n is the number of sampling points.

As local feature vector, we use an inverse Fourier transform to distinguish among objects having the same frequency. We compute a smoothed contour shape by applying an inverse Fourier transform to the low-frequency part and discarding the high-frequency part. The system calculates the difference between two feature vectors as follows:

$$SV = \sum_{i=0}^{n} \left| Rad_{sketch,i} - Rad_{model,i} \right|,$$

where n is sampling point number.

After finishing the calculation of both feature vector differences, we normalize each result by dividing it by each maximum value. The matching score is calculated as the linear sum of the normalized scores as follows:

$$Score = \alpha(1 - normFV) + \beta(1 - normSV),$$

where α, β are mean weight coefficients, *normFV* is the normalized difference of the global feature vectors, and *normSV* is the normalized difference of the local feature vectors (in our current implementation, α is 1.2 and β is 0.8). The final score of each model in the database is defined as the average between the highest and second-highest scores among the 16 views. The model panel shows the candidates of the 3-D model in descending order of matching scores.

4.2 Positioning of the Retrieved Model

The task here is to determine the position, orientation, and scale of the target model so that the model's silhouette matches the contour of the sketch. The system first needs to associate 2-D sample points of the sketch contour to the corresponding 3-D points on the model. To do this, we reuse the known relationship between the 2-D points and 3-D points of the model in the selected reference view. The 2-D sample points of the input sketch are first associated with the corresponding 2-D sample points of the reference view in the same bounding box sampling (Figure 8) and then they are associated with the corresponding points in the 3-D model. After establishing the correspondence between the 2-D points on the screen and the 3-D points of the model, the remaining task is to find the best posture parameters (position, orientation, and scale) that minimize the distance between the 2-D points and the projected 3-D points.

The detailed computation for posture estimation is as follows. Assume that the points $\mathbf{p}_1, \mathbf{p}_2, \ldots \mathbf{p}_n$ are extracted from the user's sketch and $\mathbf{v}_1, \mathbf{v}_2, \ldots \mathbf{v}_n$ are coordinates of the corresponding 3-D points of the model (Figure 9). We give the coordinates of an image point **q** as

$$\mathbf{q} = \mathbf{h}(\mathbf{Mv}), \tag{1}$$

where **v** is a world-space point that projects to **q**, **M** is a homogeneous matrix representing the combined projection and viewing transformations, and **h** is a function that converts homogeneous coordinates into 2-D image coordinates, defined by

$$\mathbf{h}(\mathbf{x}) = \left[\frac{\mathbf{x}_1}{\mathbf{x}_4}, \frac{\mathbf{x}_2}{\mathbf{x}_4} \right], \tag{2}$$

where the x_is are components of the homogeneous point x.

Our goal is to solve the following minimization problem:

$$\operatorname*{argmin}_{\mathbf{M}} \sum (\mathbf{q} - \mathbf{p})^2 . \tag{3}$$

Matrix **M** is defined by multiplying **M***object* and **M***camera*, where **M***object* is a matrix that decides a posture of a model and **M***camera* is a matrix constructed by combining various matrices subject to camera parameters. **M***camera* is given as the current camera parameter setting defined by the user, so the task is to compute **M***object*. To obtain a reasonable estimation for highly ambiguous input, we impose a couple of additional constraints to the original camera calibration method used for photographs [2]. Rotation is limited to those around the *y*-axis (the model only rotates horizontally) and scaling is limited to uniform scaling. We also use the assumption that the model is always placed on the ground or on an existing model beneath it. The resulting **M***object* matrix is defined as

$$\mathbf{M}_{\text{object}} = \begin{bmatrix} \mathbf{s}\cdot\cos\theta & 0 & -\mathbf{s}\cdot\sin\theta & 0 \\ 0 & \mathbf{s} & 0 & 0 \\ \mathbf{s}\cdot\sin\theta & 0 & \mathbf{s}\cdot\cos\theta & 0 \\ \mathbf{Tx} & \mathbf{C} & \mathbf{Ty} & 1 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \mathbf{M}_0 & 0 & \mathbf{M}_1 & 0 \\ 0 & \mathbf{M}_2 & 0 & 0 \\ -\mathbf{M}_1 & 0 & \mathbf{M}_0 & 0 \\ \mathbf{M}_3 & \mathbf{C} & \mathbf{M}_4 & 1 \end{bmatrix}^{\mathrm{T}}, \quad (4)$$

where *C* is the height of the location where the current model is to be placed. It should be zero when the new model is on the floor. Given this decomposition, we compute M by solving the following minimization problem using the Lagrange multiplier method to emphasize uniform scaling:

$$\underset{\mathbf{M}}{\mathbf{argmin}} \sum (\mathbf{q}-\mathbf{p})^2 + \lambda(\mathbf{M}_0{}^2 + \mathbf{M}_1{}^2 - \mathbf{M}_2{}^2) \cdot \quad (5)$$

We solve eq. (5) by using the Newton's method.

We also used camera calibration method to solve eq. (3) as a candidate solution when eq. (5) failed. It shows a reasonable and stable result but it does not guarantee exact uniform scaling.

5 User Study

5.1 Participants

Eight participants (one female and seven males) were recruited, mostly from the local university. All participants were frequent mouse users and seven participants had some experience with tablet PCs.

5.2 Apparatus

The user test was conducted on a Dell 8400 with a display integrated tablet. The mouse was used under the traditional menu-based UI condition and the tablet was used under the Magic Canvas UI condition. The size of the model database was 100 models.

5.3 Procedure

We compared the Magic Canvas UI to a similar in-house system with a traditional menu-based UI. The test proceeded as follows. First, we asked a professional designer to draw a rough sketch of an interior scene (Figure 10). Second, we showed the sketch to individual participants and asked them to create a 3-D scene resembling the rough sketch by combining 3-D models from a database containing several similar-looking models. Third, the designer evaluated the quality of the results. The designer was not aware of the method used for each result.

The menu-based system provides an independent mode for translation, rotation, and scaling and the user performs these operations by means of traditional direct manipulation. It also employed the same constraints used for Magic Canvas, including uniform scaling, grounding, and horizontal rotation to make the task easier. The user can put a model on another model with a simple 2-D dragging operation as in [14].

Each participant performed the task using both interfaces. Four participants used the menu-based UI first; the other four used the Magic Canvas UI first. Before testing, each participant was briefed on the operation of each interface. The menu-based UI was initiated by clicking the desired model in the model panel. The model appeared in the middle of the sketch panel and the menu appeared if the user right-clicked on a model. Users were allowed to practice before starting each experiment.

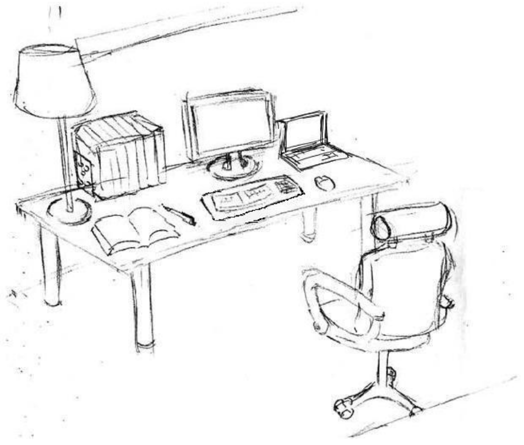

Figure 10. The concept sketch by the professional designer.

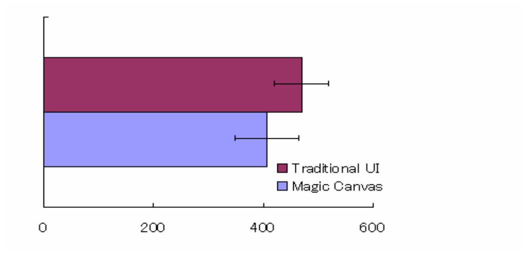

Figure 11. Completion time.

5.4 Results

Table 1 shows the task completion time. The Magic Canvas UI is faster than the traditional UI overall but we can observe that there is a case where the Magic Canvas UI is slower than the traditional UI (last two rows in Table 1). This happens when the users spend a lot of time experimenting with different possible models to place in the Magic Canvas UI and then simply reconstruct the final layout later in the traditional UI.

Table 1. Task completion time.

	Traditional UI		Magic Canvas UI	
Experiment Order	Time (sec)	Rating (1…5)	Time (sec)	Rating (1…5)
Traditional UI ↓ Magic Canvas	731	3.5	574	3
	515	4.5	398	4.5
	347	3	217	3.5
	386	3	250	4
Magic Canvas ↓ Traditional UI	472	5	365	4
	404	3	313	2
	593	4.5	709	3
	352	3	429	4

The subjective rating of the results by the designer who offered the concept sketch was 3.68 for the traditional UI and 3.5 for the Magic Canvas UI on average. This result was statistically nonsignificant, which indicates that the Magic Canvas UI was comparable to a traditional UI in terms of the quality of results.

As for the menu-based UI, all participants reported that selecting a model from the model list was very tedious and bothersome work. They frequently made mistakes in selecting menu items from the menu list. However, participants generally found that the menu-based UI was suitable for fine-tuning.

As for the Magic Canvas UI, all participants had difficulty in performing the fine-tuning because they attempted to create a scene identical to the concept sketch. They also pointed out that the wrong object appeared if the sketch was too rough and some people were confused when retrieval failed. It was particularly hard for the system to recognize very small sketches. All participants were delighted when an object appeared as they expected. Participants reported that it was advantageous to be able to examine similar candidates and that scene construction was fast when the retrieval results were satisfactory.

Some participants took the time to draw sketches in detail and some took the time to replace models and change postures for comparison with other models. The former action, drawing sketches in detail, has the significance because the sketches are remained and reused as the concept sketch. The latter actions, (e.g., replacing models) were not performed with the menu-based UI, which demonstrates that our system successfully facilitated the exploration of several alternatives compared to traditional menu-based interfaces. In addition, two participants drew detailed sketches while humming a tune, which indicated that they were enjoying the process.

6 Conclusions and Future Work

We introduced a sketch-based interactive interface and algorithms for quickly arranging multiple models from a database in a 3-D scene. The user first draws a sketch on the sketch panel and then the system retrieves a 3-D model from the database and places it in the 3-D scene so that the rendered silhouette matches the outermost contour of the input sketch. One important aspect of our system is that the user's sketch is retained in memory and serves multiple purposes (e.g., a visual reference as a 2-D image, a search queue for replacing the current model with new one), whereas editing operations in standard menu-based interfaces disappear after they are applied. One needs to perform the same sequence of operations when replacing the current model with some other model. Our findings showed that it is desirable to combine our interface with an additional interface for fine-tuning after initial placement and that it is important to provide a supplemental interface for decreasing retrieval failure.

As a next step, we are seeking to improve the retrieval algorithms using other information from sketches. We are currently applying boundary information from a sketch and are developing new algorithms that employ interior information as well. We plan to apply a weighted least squares method to give larger weights to specific areas, such as those in which the user draws overlapping strokes multiple times.

Acknowledgments

We thank all of the participants in the user study for their help.

References

[1] Eggli, L., Ching-Yao, H., and Bruderlin, B.D. Inferring 3D models from freehand sketches and constraints. *Computer-Aided Design 29* (2), 101–112, 1997.

[2] Faugeras, O. Three-dimensional computer vision: A geometric viewpoint. *MIT Press, Cambridge, MA, USA, 1993.*

[3] Fonseca, M., Barroso, B., Ribeiro, P., and Jorge, J. Sketch-based retrieval of clipart drawings. *In AVI'04: Proceedings of the ACM Press,* New York, NY, USA, 2004.

[4] Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, D., and Jacobs, D. A search engine for 3D models. *ACM Trans. Graph.* 22(1), 83–105, 2003.

[5] Gleicher, M., and Witkin, A. Through-the-lens camera control. *In SIGGRAPH '92: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques,* pp. 331–340. ACM Press, New York, NY, USA, 1992.

[6] Igarashi, T., Matsuoka, S., and Tanaka, H. Teddy: A sketching interface for 3D freeform design. *In Proceedings of SIGGRAPH '99,* pp. 409–416. ACM Press, New York, NY, USA, 1999.

[7] Ip, H.H.S., Cheng, A.K.Y., Wong, W.Y.F., and Feng, J. Affine-invariant sketch-based retrieval of images. *In CGI '01: Proceedings of the international Conference on Computer Graphics,* p. 55, IEEE Computer Society, Washington, DC, USA, 2001.

[8] Owada, S., Nielsen, F., Okabe, M., and Igarashi, T. Volumetric illustration: Designing 3D models with internal textures. *ACM Trans. Graph.* 23(3), 322–328, 2004.

[9] Pentland, A., and Kuo, J. The artist at the interface. *Vision Science Technical Report* 114, 18–26, 1989.

[10] Safar, M., Shahabi, C., and Sun, X. Image retrieval by shape: A comparative study. *IEEE International Conference on Multimedia and Expo,* pp. 141–144, 2000.

[11] Shpitalni, M., and Lipson, H. Identification of faces in a 2D line drawing projection of a wireframe object. *IEEE Trans. Pattern Anal. Mach. Intell.,* 18(19), 1000–1012, 1996.

[12] Shpitalni, M., and Lipson, H. Optimization-based reconstruction of a 3D object from a single freehand line drawing. *Computer-Aided Design,* 28(8), 651–663, 1996.

[13] Tanaka, T., Naito, S., and Takahashi, T. Generalized symmetry and its application to 3D shape generation. *Visual Computer,* 5(1–2), 83–94, 1989.

[14] Zeleznik, R.C., Herndon, K.P., and Hughes, J.F. SKETCH: An interface for sketching 3D scenes. *In Proceedings of SIGGRAPH '96,* pp. 163–170. ACM Press, New York, NY, USA, 1996.

[15] Aaron, S, Faramarz, S, and Mario, C.S. Transformation strokes. *In Proceedings of the 3rd Eurographics Workshop on Sketch-based Interface and Modeling,* Vienna, Austria, September 2006.

[16] Karthik, R., and Suyu, H. Sketch-based 3D engineering part class browsing and retrieval. *In EuroGraphics Symposium Proceedings on Sketch-based Interfaces and Modeling,* 131–138, 2006.

[17] Igarashi, T, and Hughes, J.F, A Suggestive Interface for 3D Drawing. *14th Annual Symposium on User Interface Software and Technology, ACM UIST'01,* Orlando, FL, November 11-14, 2001

[18] Tolba, O.,Dorsey, J., and Mcmillan,L.: Sketching with Projective 2D Strokes. *In Proceedings of the ACM Symposium on User Interface Software and Technology,* Asheville, NC. 1999.

[19] Cohen J,M. ,Hughes, J,F, and ZELEZNIK R.C.: Harold: A World Made of Drawings, *In Proceedings of NPAR,* 2000, pp. 83-90.

[20] Bourguignon, D., Cani, M,P, and Drettakis, G.: Drawing for Illustration and Annotation in 3D, *Computer Graphics Forum,* 20, 3 (2001), 114-122. (Proc. Eurographics '01)

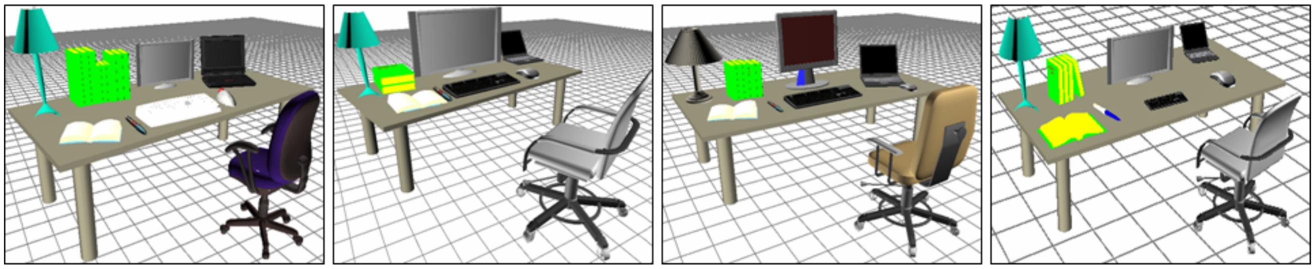

Figure 12. The part of results created by menu-based UI from the user study

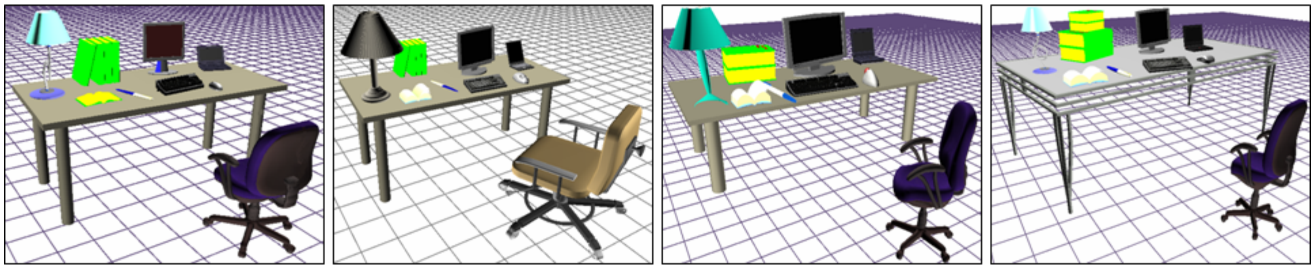

Figure 13. The part of results created by magic canvas UI from the user study

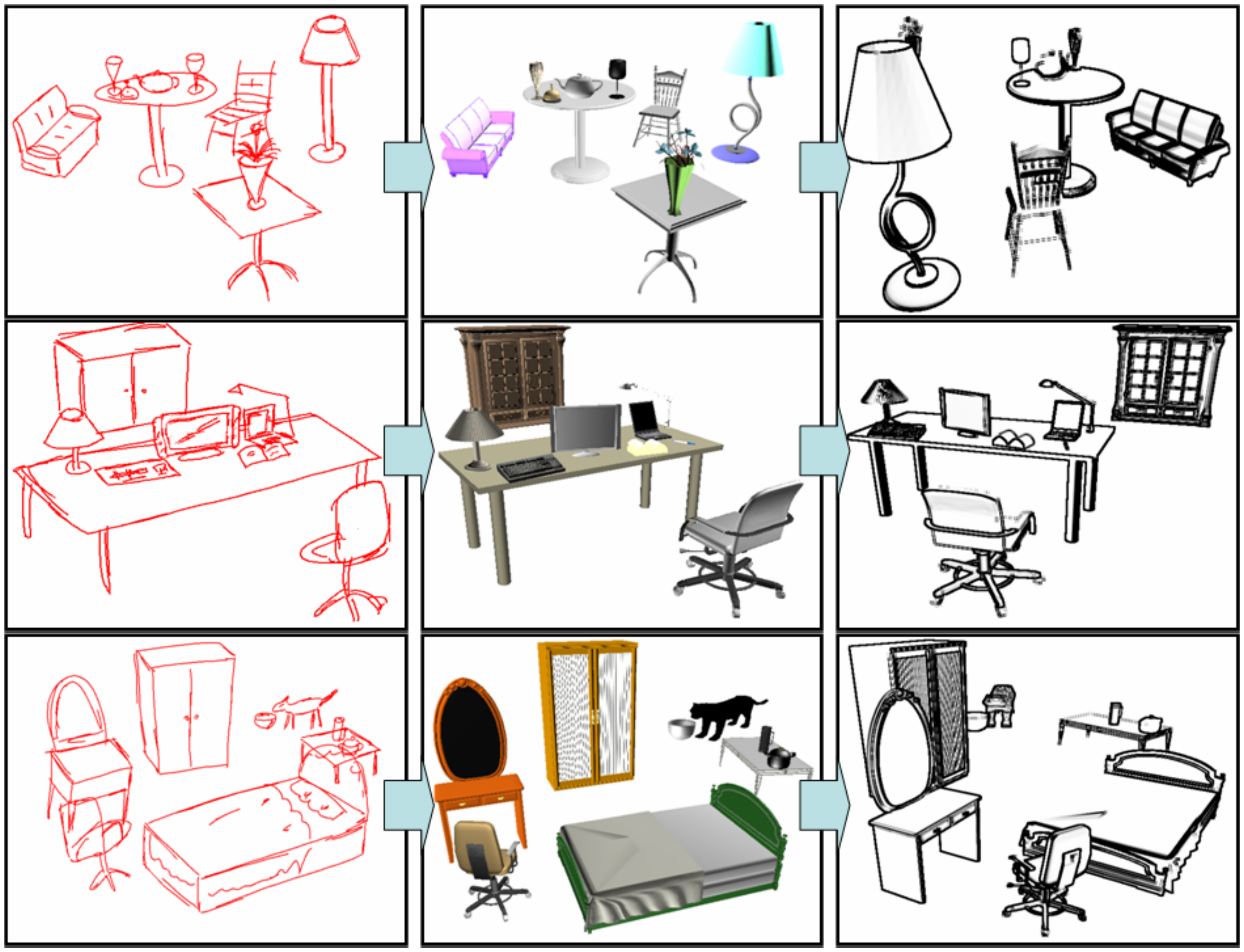

Figure 14. Examples by Magic Canvas UI. Input sketch (left), converted 3-D scenes (center), rotated scenes (right)