Proceedings of IDETC/CIE 2006
ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
September 10-13, 2006, Philadelphia, Pennsylvania, USA

DETC2006-99381

A NEW DESIGN PARADIGM BASED ON SKETCH AND RETRIEVAL

Jiantao Pu
Purdue University
pjiantao@stanford.edu

Noel Titus
Purdue University
ntitus@purdue.edu

Karthik Ramani
Purdue University
ramani@purdue.edu

ABSTRACT

To improve the efficiency and naturalness of computer-aided design (CAD), this paper introduces a new design paradigm in which a highly interactive sketch-based user interface and a design-knowledge reuse based on retrieval are combined seamlessly. Freehand sketching serves as a cognitive tool to support users' intuitive abilities by exploring the visual components of a design. Design knowledge reuse is achieved by searching the similar designs in an existing library. To demonstrate the validity of this design paradigm, we implemented two prototype systems aimed at 2D and 3D design respectively. They both use the same key components: sketch beautification module, a constraint solver, a 2D&3D search engine and a sketch-based user interface. The preliminary usability evaluation shows that our proposed design paradigm is more natural for usage and has higher efficiency than the traditional CAD systems.

INTRODUCTION

Computer Aided Design (CAD) has emerged as an important tool to facilitate the design process. As a knowledge intensive activity, the modern design process is becoming more and more complicated due to the highly limited design time and the product complexity. Consequently, the available CAD tools are often found to be inadequate and inconvenient to use, and their limitations lie in two main aspects: (1) The traditional WIMP (Windows, Icons, Menus, and Pointing Devices) interface paradigm still dominates user interfaces in most CAD applications and can not provide a designer with a natural way to express his creative ideas. Users have to be well trained and familiar with necessary functions and rules provided in a CAD system. (2) Designers do not have efficient means to harness and reuse previous knowledge and experiences. Conservative estimates suggest that more than 75% of engineering design activity comprises reuse of previous design knowledge to address a new design problem [1]. The design process has to undergo several iterations because of the lack of means to reuse past knowledge and learn from past mistakes. Also, only a limited set of design concepts are carried forward for detailed design. In the past decades, a large number of 2D drawings and 3D models in engineering fields (e.g., architecture and mechanical domains) have been produced, and they are two main ways to express and communicate design ideas. Reusing and sharing the knowledge embedded in the two representations is becoming an important way to accelerate the design process, improve product quality, and reduce costs. Significant amounts of knowledge embedded in 2D drawings and 3D models created by other personnel cannot be immediately searched and reused in design stages. Therefore, it is necessary to provide designers with a way to retrieve 3D models or 2D drawings in an intuitive way.

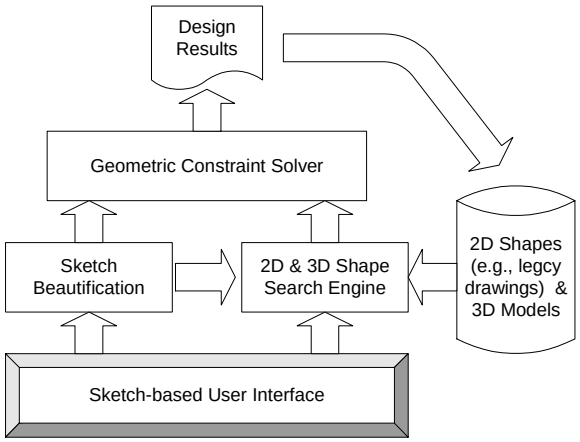


Figure 1: The framework of our design paradigm based on sketch and retrieval.

In this paper, we propose a new design paradigm which is supported by sketch-based interaction and retrieval-based

design knowledge reuse. Figure 1 shows the architecture of this design paradigm which consists of four key components: a sketch beautification module, a constraint solver, a 2D and 3D search engine and a sketch-based user interface. With the help of the freehand beautification module and the constraint solver, freehand sketches act as an intuitive and efficient human-computer interaction and are used throughout the entire design cycle. At the same time, it also serves as a tool that a designer uses to express his ideas and retrieve desired 2D and 3D shapes.

RELATED WORK

The essential relationship between sketches and the design process has been highlighted by many researchers [2-5]. For example, in Pache et al.'s survey [4], over fifty percent of the participants pointed out that they tended to use sketches before turning their ideas into CAD. The main reason arises from the fact that sketches provide the possibility to record abstract information and is very close to a designer's perception model of a product. During the design process, the sketch compensates for short-term memory limitations and at the same time supplements cognitive effort by depicting the mental imagery in a concrete form [3]. In addition, a sketch can set up a visual dialog between the designer and other group members [4]. Therefore, it will be helpful if a tool is available to transform a designer's raw sketches into a computer's precise representation. Although freehand sketch allows the designer to explore new ideas with minimal effort, the interpretation of sketches is still very difficult for computers because sketches are vague and incomplete [5].

In cognitive psychology, there have been debates about what types of knowledge representation are appropriate [6]. In Anderson's work [7], two kinds of knowledge representation were proposed: perception-based representation and meaning-based representation. The perception-based representation is more intuitive for a designer because it preserves the visual experience of a design. The meaning-based representation summarizes the most significant aspect of a design experience from the perspective of semantics. Furthermore, Hsien et al. [6] examined the roles of design knowledge from the perspective of cognitive psychology in an attempt to provide some empirical data on design knowledge. Their analysis showed that a CAD system should include a graphical representation as an entry to the database. They also pointed out that sketches can provide cues for knowledge searching because of the close connection between the perception actions and design knowledge. However, they did not explain how to perform the knowledge search based on sketches. Varejão et al. [8] proposed an ontological framework for knowledge-based design systems in order to minimize the miscommunication in computer-aided design. In their framework, five major design activities were applied in manipulating knowledge and generating end products. Also Simon et al. [9] examined the role of knowledge in next-generation product development systems and discussed knowledge representation, capture and retrieval. It addressed the need for the integration of knowledge reuse mechanism into a computer-aided design system. Based on this literature review, it is clear that the knowledge reuse is important in the process of computer-aided design. However, implementing such a system is still difficult.

Although Fonseca et al. [10] presented a preliminary study towards 3D modeling using sketches and retrieval which combined retrieval of 3D objects and expectation lists to define a new interaction paradigm based on suggestions, they did not describe their relationship clearly and the sketch is just used for the purpose of retrieval query. The advantages of sketch-based user interface are not exploited fully.

SKETCH BEAUTIFICATION

To take advantage of freehand sketch-based interaction, many methods have been proposed to parse and recognize the sketches. For parsing purpose, the work outlined in [11, 12] tried to explore the interactive nature of sketching such as the stroke direction and speed. However, the sketching activity has to be recorded in real time. The perturbation of a user's hand will lead to obvious changes in curvature and speed, and thus result in incorrect segmentation. To overcome such limitations, many other approaches were proposed such as the template-based method [13] and the Bayesian-based statistical model [14]. Unfortunately, these methods are still not capable of handling the parsing problem robustly. Then some other approaches [15, 16] imposed constraints on users' behaviors. For example, a user was required to explicitly indicate the intended portioning of the ink, i.e., each pen stroke must represent a single shape, such as a single line segment or arc segment. Despite their simplicity, the single-stroke requirement usually resulted in a less natural interaction. Following sketch parsing, the next important step is the recognition of the parsed result. Approaches interpreting individual geometric shapes include HMM-based algorithm [17], Zernike moment features [18] and multi-layer image recognition scheme [19]. However, most recognizers are either hand-coded or require large sets of training data to reliably learn new symbols. They still could not assure an ideal accuracy. In contrast, we proposed a new algorithm [20] to parse and recognize sketches. The parsing procedure is independent of the sketching speed and curvature, stroke-order, -number, and -direction, as well as invariant to rotation, scaling, and translation of strokes. It can also be used to recognize composite shapes.

To parse a stroke into independent primitives, we propose a circle-scanning strategy [20], in which multiple circles are used to scan the sketches by changing the radius of each circle progressively. When two neighbor intersected points between a scanning circle and the sketch are close enough, the shared point between the two respective intersected line segments is regarded as the critical point or segment point. As for the detailed explanation, please refer to our paper [20]. Figure 2(a)~(c) show the segmentation examples of some 2D-like sketches, while Figure 2(d)~(f) show the segmentation examples of the 3D-like sketches.

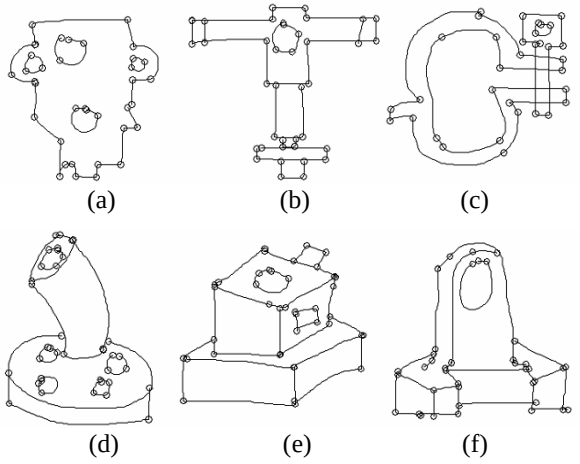


Figure 2: Some sketch parsing examples.

To recognize a sketched shape, we use a template-based shape similarity method. Given a parsed segment, we compute the similarities between this segment and all predefined geometric primitive such as line, circle and arc. The most similar primitive can tell what kind of primitive the parsed segment represents. As for the introduction of the shape similarity computation, refer to Section 4.3. In practice, this approach can also be used to recognize a sketched composite.

Once the primitives in the sketches are recognized, they can be beautified. However, during the sketching process, it is difficult for users to sketch precisely, and over-sketching or under-sketching is unavoidable. Simply converting the segments into the corresponding regular primitives is not enough. To handle these problems, we assume that users usually start and end their sketches at a certain geometric entity. In this way, we can use the nearest-neighbor principle to handle these drawbacks: find the nearest entity on which the starting point or the ending point of a sketch are located and the intersecting point between the two entities is the right starting or ending point.

Another problem is that an arc can be sketched differently. Frequently the proposed segment method will parse a sketched arc into several parts. To handle this issue, we propose a rule: if two neighboring parsed segments are recognized as arcs and their angles have the same direction, then they will be merged as one arc.

Finally, during the beautification, we have to determine the parameters for each geometric entity. For a line, we need to set the starting point and ending point. We can adopt the starting and ending points of the sketch segment as the initial starting and ending points of a line. Later the two points can be adjusted after over- and under-stroke process. For an arc or circle, we select three points on the sketch segment and use them to compute the radius and the center of the arc or circle since three points determine an arc or circle. The three points can be selected in a simple way: the two ending points and the mid point of the parsed primitive.

2D & 3D SHAPE SEARCH ENGINE

2D legacy drawings and 3D models are important visual media to represent design ideas. Reusing and sharing the knowledge embedded in the two representations is becoming an important way to accelerate the design process, improve product quality, and reduce costs. Until now, many methods have been proposed to retrieve 3D models from a database. These methods can be classified into four categories: (1) feature-vector based methods [21], (2) statistics-based methods [22], (3) topology-based methods [23, 24], and (4) image-based methods [25, 26]. In contrast to the attentive efforts in the 3D model retrieval, less effort [16] has been made to 2D drawing similarity. To make use of the knowledge embedded in these two media, we developed a 2D & 3D shape search engine called ShapeLab by which designers can retrieve 3D models or 2D drawings using different queries, such as freehand sketches, 2D drawings, or 3D models. The query process is similar to the process where engineers express their shape concepts on a piece of paper, and no special training is required for novices. Compared to the method presented in [25, 26], ShapeLab only needs three views that are determined automatically to represent the 3D shape. Users can not only state preferences by specifying weight values for views, but can also do editing operations on each of the three views obtained from the retrieved models. The whole retrieval process forms a refining loop through which users can search 3D models or 2D drawings in a coarse-to-fine way. ShapeLab is composed of four key components: (1) pose determination for 3D models: bridging the space gap between 2D drawings and 3D models by finding three intuitive orthogonal orientations for 3D models; (2) 2D orthogonal drawing-like view generation from 3D models: representing a 3D model by three orthogonal views along the pose orientations; (3) similarity measurement between 2D drawings: finding 2D drawings and 3D models with desired geometric shapes; and (4) freehand sketch-based user interface: providing a natural way for designers to express shape intentions.

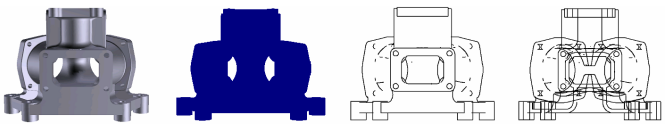
3D Shape Representation

In our previous work [27] we have proposed a method to compute the pose of a 3D model by finding the orthogonal orientations with maximum virtual contact area (VCA). The basic idea stems from the fact that the orthogonal directions that have the maximum VCA (Virtual Contact Areas) are the major principal axes of the 3D model and provide a stable pose. The key step in obtaining the principal axes is to determine the polygons of a 3D object that have the same normal and lie in the same plane. VCA is defined as the bounding area formed by polygons that have the same distance from a predefined point and have the same normal. To obtain the direction along which the VCA is the maximum, we need to find all polygons that have the same normal direction and the same distance to the mass center. The direction that gives the maximum VCA is the first principal axis $\mathbf{b}^u$ of the 3D object orientation. To get the next principal axis $\mathbf{b}^v$ of an object orientation, we find the

normal that satisfies two conditions: (a) is orthogonal to the first principal axis; and (b) has maximum area. The third axis can then be obtained by performing the cross product between $\mathbf{b}^u$ and $\mathbf{b}^v$:

$$\mathbf{b}^w = \mathbf{b}^u \times \mathbf{b}^v \quad (1)$$

To depict a 3D model precisely using 2D views, we abstract the 3D shape as multiple levels of detail as illustrated in Figure 3. It can be seen that the contour level reflects its global shape by which a user can "guess" the true object to some extent. The silhouette level conveys more shape details using a few more simple sketches compared to the contour level. When the detailed shape information is not important, the silhouettes are enough to differentiate two similar objects more confidently as compared with the contours. The third level contains the complete information, including the visual appearance and the occluded structure, by which a user can figure out its shape precisely. In practice, especially in engineering fields, there are a lot of models that look similar from visual appearance but still have different inner structures. Therefore, we need to consider the complete details of the object. However, the projected shapes at the three levels are different from different view points. With the help of the pose determination method, we project a 3D shape onto the six faces along the principal axes to represent the MLD. At the contour level, there are three different views along the principal axes; at the silhouette level, there are six different views; and at the full level, we use the traditional drawing-like views to represent the drawing level, and there are three different views along the principal axes.



(a) (b) (c) (d)

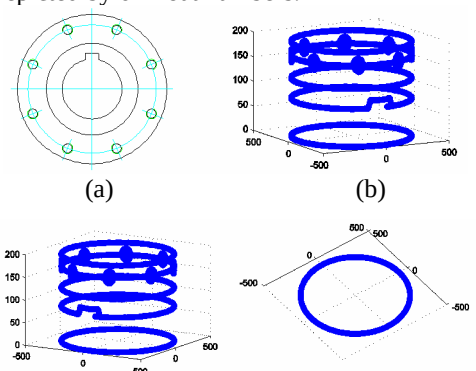
Figure 3: Multiple levels of detail: (a) a 3D model (b) contou r level; (c) silhouette level; (d) drawing level.

2D Shape Descriptors

In [28], we have proposed two methods to retrieve 2D drawings by measuring their shape similarity. The first approach represents a drawing as a spherical function by transforming it from 2D space into 3D space and then employs a fast spherical harmonics transformation to get a rotation invariant descriptor. The second method represents the shape of a 2D drawing from the statistics perspective as a distance distribution between pairs of randomly sampled points. Both the representations have many valuable advantages: invariant to affine transformation, insensitive to noise or cracks, simple, and fast.

A 2D analog of the spherical harmonics was used in [25] to extract a series of rotation invariant signatures by dividing a 2D silhouette shape into multiple circular regions. However, this method has two major limitations as mentioned in [28]: one-to-multiple correspondence and instability caused by shape perturbation. In order to overcome these limitations and thus obtain a set of robust rotation invariant signatures for a 2D shape, we propose a strategy called 2.5D spherical harmonics representation [28] which can extract a series of rotation invariants by transforming a 2D shape from 2D space into 3D space uniquely. The name "2.5D" arises from the fact that a 2D shape is represented in a 3D space.

The basic steps can be described as follows. First, given a 2D shape, a sphere is determined and its equator plane encloses the 2D shape. Second, a set of rays located in the equator plane of the sphere are uniformly shoot out from the sphere center. Thus the interested points between the rays and the 2D shape are regarded as an approximation of the 2D shape. Third, to represent the intersected points using a one-to-one mapping spherical function, these points are projected onto a cylinder whose basis plane is the same as the equator plane of the sphere. Each point has a different height that is equal the distance between the sphere center and this point. Consequently, a 2D shape is uniquely transformed into a 3D spherical representation. We name this process a 2.5D transformation. Figure 4 shows an example of this transformation for a 2D shape example. From this example, we notice that the geometric information is represented clearly in 3D space along the surface of a cylinder. Finally, to obtain the rotation-invariant, we use the fast spherical harmonics transformation method [29] in which a spherical function of bandwidth B is sampled on the 2B-many Chebyshev points. These sampled points form a 2B×2B equiangular grid along the longitude and latitude of a sphere. In our implementation, B is equal 64. It means that a given 2D shape is depicted by 64 float numbers.



(a) (b) (c) (d)

Figure 4: An example of 2.5D spherical harmonics representation: (a) is a 2D shape; and (b), (c), and (d) show the 3D representation of the drawing from different perspectives.

We also proposed another 2D shape descriptor [28] which represents a 2D shape as a distance histogram by sampling enough random points uniformly on the 2D shape. Thus the similarity between two 2D shapes can be measured by computing distance distributions formed by the random points.

Given two 2D shapes, their similarity is measured by computing the Minkovski distance L_n between their histograms, i.e., $\mathbf{H}_1$ and $\mathbf{H}_2$:

$$W(H_1, H_2) = L_n(H_1, H_2) = \sqrt[n]{\sum_{i=0}^{h} (H_1(i) - H_2(i))^{1/n}} \qquad (2)$$

where h is the dividing number of a histogram. The smaller W means more similarity between two histograms.

Since a 3D model is represented by three orthogonal drawing-like views, a procedure is needed to intuitively find the correspondence between drawing-like views from different models. To simplify this process, we use the minimum values between all possible view-pairs to represent their similarity.

Given the two shape similarity methods, it is natural to try combining the two shape descriptor together to achieve better performance. We propose applying a weight value to each of the two shape similarity methods and using their combined confidence to measure overall similarity.

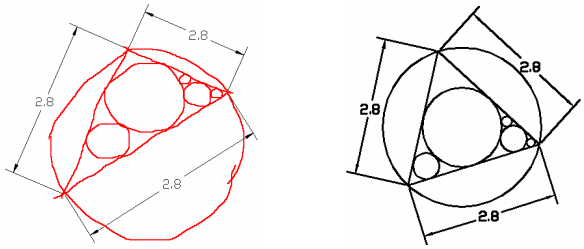
GEOMETRIC CONSTRAINT SOLVER

Geometric constraint solving is becoming a powerful tool to express and reason the design intent of geometric models by specifying or changing the constraints. In the engineering domain, a large body of literature on constraint solving addresses the problem with a variety of approaches. The proposed methods were classified into four categories: (1) Numerical solvers [30]; (2) Symbolic solvers [31]; (3) Rule-based solvers [32]; and (4) Graph-based solvers [33]. Although a lot of effort has been made to address this issue, the maximum division of a geometric constraint system into subproblems is still a difficult problem. In [34], we propose a graph reduction approach by which most geometric constraint satisfaction problems can be solved analytically. The basic idea is representing all geometric constraints as a graph structure and use the freedom analysis method to reduce the graph. In contrast with previous graph-based methods, an interface-open and –reconstruction strategy is introduced to maximally reduce the graph so that a constraint system can be solved analytically. The reduction process is recorded by a tree structure called a reduction tree. Once the reduction tree is generated, an analytical solution will be obtained by traversing the tree from the bottom up. By assigning proper weight to each constraint, we can handle over- and under-constraint systems automatically. Although the proposed method is implemented for a 2D case, it can be extended to the 3D case in a similar way.

To determine the position of a primitive, this primitive must be fixed with respect to a known primitive, i.e., the degree of freedom of this primitive is zero with respect to the known primitive. Therefore, it is natural to apply freedom analysis to the geometric constraint problem. The degree-of-freedom (DOF) of a primitive is the number of independent parameters to describe its position and shape. For example, a point has 2 DOFs because it can be described by its coordinate (x, y); a line has 2 DOFs because it can be described by an intercept distance and an angle with x axis; a circle has 3 DOFs because it can be described by its center (x, y) and a radius r. In particular, a super-primitive (or super-node) has 3 DOFs since it acts like a rigid body. In contrast, there is a concept called degree-of-constraint (DOC), which is the decreased number of DOF when a constraint is imposed. In 2D space, a DOC between any two primitives can be 1, 2, or 3.

For a geometric constraint system, by enumerating all kinds of constraints between points, lines, circles, and super-primitives, we summarize the constraint patterns in a GCG (Geometric Constraint Graph) as four types. The number beside a node represents the DOF, and the number beside an edge represents the DOC. The four patterns cover all constraint cases. Under these patterns, the position of a node can be determined with respect to the position of another node. In other words, if there is any one constraint pattern (CP) that satisfies one of the predefined patterns, we can determine the relative position between the primitives or super-primitives that the nodes represent. Therefore, the four constraint patterns constitute the basis of graph reduction operation.

This reduction process can be recorded in a binary-tree structure called a reduction tree. Each tree represents a set of primitives whose positions can be determined with respect to the basis node. Due to the introduction of the interface-open strategy, there are common nodes between these reduction trees. With the help of these common nodes, these reduction trees can be correlated together, i.e., the position of a reduction tree with respect to that of another tree will be determined. In Figure 5, some examples of beautification driven by constraints are presented. When these drawings are finished and the constraints are imposed, they will be beautified automatically.



(a) (b)

Figure 5: A sketch beautification example driven by geometric constraint.

In order to handle over- and under-constrained problems, we propose a priority based strategy: all constraints are classified into different priorities. Each constraint is assigned a priority level between 0~5: “0” means the highest priority, and “5” means the lowest priority. On the whole, explicit constraints have a higher priority than implicit constraints do. When there are several candidates that satisfy the reduction conditions, we will check their priority and only reduce the node with the highest priority. Generally, we can obtain more constraints than needed since many implicit constraints are infered. Using the

priority strategy, the over- and under-constrained problems can be solved transparently.

Since the constraint-based beautification can provide designers more flexibility than traditional interaction paradigms. Freehand sketches generally contain some implicit constraints. To detect implicit constraints, we propose a method named *Relative Shape Histogram* (RSH) to check the relationship between two geometric primitives. RSH is similar to the shape histogram method described in Section 4.3.2. RSH has the same sketch representation, the same sampling strategy, and the same shape function. The only difference is that RSH only considers the ***D*2** distances between point pairs that are sampled from different primitives, i.e., point pair (x_i, y_i) and (x_{i+1}, y_{i+1}) are located on different primitives for RSH.

IMPLEMENTATION

The methods mentioned in this paper have been implemented as dynamic link libraries (DLLs) using C++. With the help of these DLLs, we implemented two prototype systems targeting at computer-aided design. One was incorporated into AutoCAD platform, aimed at 2D computer-aided design. The other was incorporated into Unigraphics, aimed at 3D computer-aided assembly. They had the same architecture shown in Figure 1, and the interactions between the key modules can be illustrated by Figure 6. During the design process, a user can sketch her ideas using a pen without turning to the traditional interaction ways such as keyboard and mouse. The sketched result can be used in two ways: retrieve similar design schemes available in database and be beautified for precise input. With the help of the constraint solver, a formal design is created. Users are allowed to switch between these modules flexibly according to their needs.

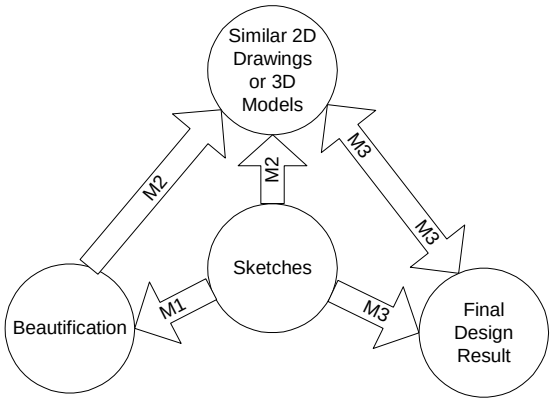


Figure 6: Interactions between the key modules introduced in this paper: (1) M1: sketch beautification module; (2) M2: 2D and 3D shape retrieval module; and (3) M3: geometric constraint satisfaction module.

In addition, we also provided several feedback ways for users to interactively refine retrieval results:

(1) Weight Value Adjustment for Orthogonal Views: In this paper, a 3D model is described by three orthogonal views. Different views reflect certain shape characteristics from different perspectives. To find similar shapes with certain features, users are allowed to emphasize certain views by adjusting their weights. The larger the weight value is, the more important the view will be. Users can repeat this process to refine the searched results.Geometric constraint solving is becoming a powerful tool.

(2) Initial Sketch Input Modification: In some cases, just adjusting the weight value is not enough because the initial sketch is not sufficient to represent the desired models and sometimes users make mistakes. Enabling a user to edit the initial sketch input is an efficient way to refine the search results, especially when the returned results can be displayed dynamically as the sketch is modified. Users can modify the input sketches by examining the retrieved results.

(3) Retrieved Result Modification: As mentioned above, retrieved results are used to help users to check and modify the improper input. We propose a third approach to edit the views of the retrieved 3D models or 2D drawings to improve and refine the retrieval results. In contrast with the previous feedback mechanism, this feedback mechanism allows uses to edit the views of the retrieved models.

Generally, it is not easy for users to find proper weight values or modify the sketches just within one time. Users usually have to repeat this process several times. To accelerate this process, we also provide a dynamic retrieval mechanism. A user's behavior is monitored by the system all the time. During the feedback-based interactions, once an operation is finished, the retrieval results will be dynamically refreshed so that a user can experience the impact of his operation on the retrieval results. In this way, users can get instant feedback and adjust their operations efficiently.

PRELIMINARY EVALUATION

In this section, we present the preliminary result about the usability of the proposed design paradigm supported by these modules. Usability testing focuses on determining if a software is easy to use and learn, and contains the functionalities that a user desire, thus making this system effective. In this paper, we use the two applications mentioned in Section 6 as test beds to evaluate the usability of our proposed design paradigm. Nine test participants were selected and asked to create given 2D legacy drawings and 3D assemblies in two ways: traditional WIMP user interface and sketched-based user interface. We designed a questionnaire for participants' experiences about the two kinds of interactions. The purpose was to gain an overall comparison between the traditional computer-aided design system and our design paradigm supported by sketch and retrieval. All the participants are students from mechanical engineering. They had knowledge of the design process. However, they were not fully familiar with AutoCAD and Unigraphics platforms. Therefore, we gave all participants a brief tutorial about the necessary commands before stating the test, including some guidance about our prototype system. More detailed information about the evaluation test will be introduced

in a future publication. In this paper, we only summarize the most significant results and conclusions obtained from the test.

The evaluation is done into two parts. The first part is the subjective evaluation based on the questionnaire. As shown in Table 1, the numbers of the responses towards different testing items are summarized. It demonstrated that most participants tended to think the design paradigm based on sketch and retrieval is much easier to learn and use than the design paradigm based on traditional WIMP user interface. However, as for the frustration caused by errors, we found it was more natural for participants to accept the errors caused by an accident of their faults rather than the errors caused by computer recognition, such as parsing error and primitive recognition errors. Even a small number of recognition errors leads to user frustration.

Table 1: Subject evaluation and comparison based on questionnaire

Subject Evaluation		*Excellent*	*Very Good*	*Good*	*Poor*	*Very Poor*
Ease of Learning	WIMP	0	2	4	3	0
	Sketch&Retrieval	6	2	1	0	0
Ease of Using	WIMP	1	1	4	3	0
	Sketch&Retrieval	4	3	2	0	0
Ease of Editing	WIMP	3	2	2	2	0
	Sketch&Retrieval	3	2	4	0	0
Frustration Toward Errors	WIMP	0	0	2	3	4
	Sketch&Retrieval	1	2	4	2	0

The second part is a quantitative comparison between the efficiency of the popular CAD design systems such as AutoCAD and Unigraphics. The result is shown in Table 2.

Table 2: Quantitative efficiency comparison between the popular CAD design systems and our proposed design paradigm.

Platforms		AutoCAD	2D Prototype System	Unigraphics	3D Prototype System
Average Time to Learn (Hours)	<0.5	1	9	2	9
	1~2	5	0	4	0
	3~4	2	0	3	0
	4~5	1	0	0	0
	>5	0	0	0	0
Average Time for a Task (Hours)	<0.5	2	7	1	6
	0.5~1.0	5	2	6	3
	1.0~1.5	2	0	1	0
	1.5~2.0	0	0	1	0
	>2.0	0	0	0	0

It can be seen that it is much easier for participants to be familiar with the usage of our proposed design paradigm than the traditional design paradigm based on WIMP user interface. The efficiency of our proposed design paradigm is also higher than the traditional design systems. With the help of the questionnaire, we found the reason lies in the fact that the commands for the sketch & retrieval based design are naturally implied in users' intuitive actions, while the commands for the traditional CAD systems must be memorized in advance.

Finally, when we asked for some comments from the participants, they suggested that they would rather combine the two kinds of design paradigms together and believed that such combination would become more natural and efficient. Also the traditional design systems are still more robust than the proposed systems because they are more tolerant to their own errors rather than the computer's errors.

CONLCUSION

In this paper, we introduced a new design paradigm which is supported by four key modules: a sketch beautification module, a constraint solver, a 2D&3D search engine and a sketch-based user interface. The motivation is overcoming the limitations of the traditional computer-aided design such as unnaturalness and inefficiency in current CAD systems which are becoming more and more serious as the CAD systems become more and more complex. The proposed design paradigm has two distinct characteristics: (1) freehand sketch is used in the whole design cycle; and (2) the design knowledge implied in 2D legacy drawings and 3D models is reused conveniently by performing search operations. Usability evaluation demonstrated that this design paradigm is more natural and efficient than the traditional WIMP paradigm.

ACKNOWLEDGMENTS

Put acknowledgments here.

REFERENCES

1. Ullman, D.G., "The Mechanical Design Process", 2nd Edition, McGraw-Hill, NY, 1997.
2. Beryl, P., and Mark, A., "Computer-aided Sketching to Capture Preliminary Design," In Proceedings of ACM International Conference Proceeding Series on Third Australasian conference on User interfaces, Vol.7, pp.9-12, Melbourne (Australia), 2002.
3. Pache, M., and Lindemann, "Sketching in 3D Human Behaviour in Design," Springer Berlin, 2003.
4. TVERSKY, B., "What Does Drawing Reveal about Thinking," In Proceedings of Visual and Spatial Reasoning in Design, Cambridge (USA), 1999.
5. van Dijk, C. G. C., "New Insights in Computer-aided Conceptual Design," Design Studies, 16(1):62-80, 1995.
6. Tang, H.H., and Gero, J.S., "Roles of Knowledge While Designing and Their Implications for CAAD," Proceedings of CAADRIA 2001, Sydney (Australia), pp. 81–89, 2001.
7. Anderson, J. R., Cognitive Psychology and its Implications (4th ed.). W.H. Freeman and Company, New York, 1995.
8. Varejao, F.M., Menezes, C. S., Garcia, A.C.B., Souza, C.S., and Fromherz, M.P.J., "Towards an Ontological Framework for Knowledge-Based Design Systems," Proceedings of 6th International Conference on Artificial Intelligence in Design (AID'00), Worcester (USA), pp.55-75, 2000.
9. Simon, S., Ram, D.S., and William, C.R., "The Role of Knowledge in Next-generation Product Development Systems," The Journal of Computing and Information Science in Engineering, vol.1(1): 3-11, 2001.

10. Fonseca, M.J., Ferreira, A., and Jorge, J.A., “Towards 3D Modeling using Sketches and Retrieval,” In Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBM'04), Lisboa (Portugal), pp.127-136, 2004.
11. Sezgin, T.M., Stahovich, T., and Davis, R., “Sketch Based Interfaces: Early Processing for Sketch Understanding,” In Proceedings of the 2001 Workshop on Perceptive User Interfaces, Orlando (USA), pp.1-8, 2001.
12. Calhoun, C., Stahovich, T.F., Kurtoglu, T., and Kara, L.B., “Recognizing Multi-Stroke Symbols,” In Proceedings of the 2002 AAAI Spring Symposium - Sketch Understanding, Palo Alto (USA), pp.15-23, 2002.
13. Hse, H., Shilman, M., and Newton, A.R., “Robust Sketched Symbol Fragmentation Using Templates,” In Proceedings of the 9th International Conference on Intelligent User Interface, Funchal (Portugal), pp.156-160, 2004.
14. Shilman, M., Pasula, H., Russell, S., and Newton, R., “Statistical Visual Language Models for Ink Parsing,” In Proceedings of AAAI Spring Symposium on Sketch Understanding, Palo Alto (USA), pp.126–132, 2002.
15. Landay, J.A., and Myers, B.A., “Sketching Interfaces: Toward More Human Interface Design,” *IEEE Computer*, 34(3): 56-64, 2001.
16. Forbus, K., Lockwood, K., Klenk, M., Tomai, E. , and Usher, J., “Open-domain Sketch Understanding: The nuSketch Approach,” In Proceedings of the AAAI Fall Symposium on Making Pen-based Interaction Intelligent and Natural, Washington, D.C. (USA), pp.58-63, 2004.
17. Sezgin, T.M., and Davis, R., “HMM-Based Efficient Sketch Recognition,” In Proceedings of the 10th International Conference on Intelligent User Interfaces, San Diego (USA), pp.281-283, 2005.
18. Hse, H., and Newton, A.R., “Sketched Symbol Recognition Using Zernike Moments,” In Proceedings of 17th International Conference on Pattern Recognition (ICPR'04), Cambridge (UK), pp.367-370, 2004.
19. Kara, L.B.., and Stahovich, T.F., “An Image-Based Trainable Symbol Recognizer for Sketch-Based Interfaces,” In Proceedings of AAAI Fall Symposium Series 2004: Making Pen-Based Interaction Intelligent and Natural, Arlington (USA), pp.99-105, 2004.
20. Pu, J.T., and Karthik, R. “Toward Freehand Sketch Beautification Driven by Geometric Constraint”, submitted to ACM Transaction on Computer Graphics.
21. Elad, M., Tal, A. and Ar, S., “Content Based Retrieval of VRML Objects: An Iterative and Interactive Approach,” In Proceedings of 6th Eurographics Workshop on Multimedia 2001 (Manchester, UK, 2001), pp.107-118, 2001.
22. Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D., “Shape Distribution,” ACM Transactions on Graphics, 21(4): 807-832, 2002.
23. Hilaga, M., Shinaagagawa, Y., Kohmura, T. and Kunii, T.L., “Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes,” in Proceedings of SIGGRAPH 2001(Los Angeles, USA, 2001), pp 203–212, 2001.
24. Siddiqi, K., Shokoufandeh, A., Dickinson, S., and Zucker, S., “Shock Graphs and Shape Matching. Computer Vision,” Vol.35 (1): 13-20, 1999.
25. Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, D., and Jacobs, D., “A Search Engine for 3D Models,” ACM Transactions on Graphics, 22(1): 83-105, 2003.
26. Chen, D.Y., Tian, X.P., Shen, Y.T., and Ouhyoung, M., “On Visual Similarity Based 3D Model Retrieval,” Computer Graphics Forum (Eurographics’2003), 22(3):223--232, 2003.
27. Pu, J.T., and Karthik, R., “An Automatic Drawing-like View Generation Method from 3D Models”, In Proceedings of ASME IDETC/CIE 2005, 25th Computers and Information in Engineering (CIE) Conference, Long Beach (USA), pp.301-320, 2005.
28. Pu, J.T., and Karthik, R., “On Visual Similarity based 2D Drawing Retrieval”, accepted by the Journal of Computer Aided Design.
29. Healy, D., Kostelec, P., Moore, S., “FFTs for the 2-Sphere--Improvements and Variations,” Journal of Fourier Analysis and Applications, 9(4): 341 – 385, 2003.
30. Lamure, H., and Michelucci, D., “Solving Geometric Constraints by Homotopy,” *IEEE Transaction on Visualization and Computer Graphics*, *2(1)*, 28-34, 1996.
31. Kondo, K., “Algebraic Method for Manipulation of Dimensional Relationships in Geometric Models,” *Computer Aided Design*, *24(3)*, 141-147, 1992.
32. Verroust, A., Schonek, F., and Roller, D., “Rule-oriented Method for Parameterized Computer-aided Design,” *Computer Aided Design*, *24(10)*, 531-540, 1992.
33. Hoffmann, C. M., Lomonosov, A., and Sitharam, M., “Finding Solvable Subsets of Constraint Graphs,” *Lecture Notes in Computer Science*, *1330*, 163-197, 1997.
34. Pu, J.T., and Karthik, R. “An Analytic Approach to Geometric Constraint Solving”, submitted to the Journal of Artificial Intelligence for Engineering Design, Analysis and Manufacturing.