Sketch Matching on Topology Product Graph

Shuang Liang, Jun Luo, Wenyin Liu, and Yichen Wei

Abstract—Sketch matching is the fundamental problem in sketch based interfaces. After years of study, it remains challenging when there exists large irregularity and variations in the hand drawn sketch shapes. While most existing works exploit topology relations and graph representations for this problem, they are usually limited by the coarse topology exploration and heuristic (thus suboptimal) similarity metrics between graphs. We present a new sketch matching method with two novel contributions. We introduce a comprehensive definition of topology relations, which results in a rich and informative graph representation of sketches. For graph matching, we propose topology product graph that retains the full correspondence for matching two graphs. Based on it, we derive an intuitive sketch similarity metric whose exact solution is easy to compute. In addition, the graph representation and new metric naturally support partial matching, an important practical problem that received less attention in the literature. Extensive experimental results on a real challenging dataset and the superior performance of our method show that it outperforms the state-of-the-art.

Index Terms—Sketch matching, topology relations, similarity metrics

1 Introduction

SKETCHING has long been considered as one of the most natural and flexible interfaces in human computer interaction. It is useful in creative tasks such as drawing, writing, designing, and so on. For decades, a lot of research efforts have been spent on enhancing computers' capabilities to better assist such tasks, allowing for quick drafting, informal communication, ambiguous expression, and instant feedback. Recently, such needs are greatly boosted by the fast explosion of personal touch devices and extended from professionals to amateurs.

Among the key capabilities of a sketch interface, arguably *sketch recognition/retrieval* is the critical one, that is, the capability of converting the user's raw input into his/her intended shape [1]. Usually, this is realized by comparing the input raw sketch against a pre-built sketch database and finding the most similar one. In this process, the essential problem is to compare and measure the similarity of two sketches, called *sketch matching*. It is the fundamental problem in almost all sketch based tasks, such as sketch-based design [2], quick diagramming [3], and sketch-based retrieval [4], [5], [6].

The basic features in sketch matching are *geometry properties* of sketched shapes, such as position, size, and angle. Yet, only relying on such features to match hand drawn sketches is not reliable because they are sensitive to variations in the free drawing process, such as different drawing styles, unstable stroking, and indefinite shape position/size. Such uncertainties are especially significant for general amateur users. For example, the sketches in Fig. 1 are drawn by four different users for the same template shapes. Such drawings are quite irregular and therefore hard to match.

- *S. Liang is with the School of Software Engineering, Tongji University, Shanghai, China. E-mail: shuangliang@tongji.edu.cn.*
- *J. Luo is with Huawei Noah's Ark Laboratory, Hong Kong. E-mail: luo.jun1@huawei.com.*
- *W. Liu is with Multimedia Software Engineering Research Center, City University of Hong Kong, Hong Kong. E-mail: liuwenyin@gmail.com.*
- *Y. Wei is with Microsoft Research Asia, Beijing, China. E-mail: yichenw@microsoft.com.*

Manuscript received 6 Feb. 2014; revised 20 Oct. 2014; accepted 30 Oct. 2014. Date of publication 9 Nov. 2014; date of current version 6 July 2015.
Recommended for acceptance by M. S. Brown.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2014.2369031

A good sketching interface should be able to ignore irrelevant local details but focus on capturing the more global and perceptual semantics. For this purpose, all the state-of-the-art sketch matching methods [2], [4], [7], [8], [9], [10], [11] exploit the *topology relations* between sketch primitives, such as parallelism and intersection. Such relations are more invariant to geometrical variations and capture the sketch semantics more globally. They are therefore useful to match perceptually similar sketches that may vary in details. While such methods have been shown effective, in general they still suffer from two problems. First, their topology relations are usually coarse and limited in expressive power. Second, while they all use graph to represent the relations, they do not use exact graph matching because it is NP-hard and computationally too expensive. Instead, heuristic and approximate similarity metrics are used, which are usually hard to understand and suboptimal.

Consequently, a practical drawback in previous sketch matching methods is that they do not work well for *partial matching*, that is, the user drawn sketch is incomplete. The problem of supporting partial matching has received relatively less attention in the literature. However, it is becoming more important for a modern sketch interface when more amateur users are involved with the popularization of personal touch devices. Such users are usually clumsy in drawing their whole ideas and can only depict the most representative parts. The capability of matching the whole sketch from an incomplete input could greatly save users' effort, allow them to focus on their real creative work, and make the system more effective.

In this work, we present a new method that effectively addresses the issues mentioned above. Our main contribution is a novel *topology product graph* based sketch matching approach. The product graph combines two topology graphs that encode the topology and geometry information of the two sketches to match. While this concept is derived from general graph theory, this work is the first to exploit "product graph" for sketch matching. We propose effective graph construction methods so that the resulting product graph retains the full correspondence information for matching two graphs. This motivates a new sketch similarity metric that is intuitive, allows an exact solution and supports partial matching in nature. To better exploit the topology information, we present a novel *fine-grained topology graph* that is defined on a comprehensive and rich set of topology relations. Such topology graphs capture the fine structures of complex sketches and they are used to create the product graph.

Observing that previous commonly used datasets are usually too simple and inadequate to reflect the real challenges in sketching, we create a new hand drawn sketch dataset that is more comprehensive in shape complexity, diversity, magnitude and user variations than previous ones. Extensive experimental results on the dataset verify that the proposed approach clearly outperforms the state-of-the-art, in both complete and partial matching. The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 describes our approach. Section 4 introduces the new dataset and reports the experimental results. Section 5 concludes our work.

2 Review of Related Work

We first review related work from three aspects of sketch matching: *preprocessing*, *representation* and *matching*. We then briefly discuss *partial matching*.

Raw input of hand-drawn sketches consists of noisy and inaccurate strokes. Such data cannot be directly used for sketch matching. They are usually refined and decomposed into a few basic primitives [12], [13], which are more stable representations to different user styles. Such preprocessing has been well studied in the literature and we use the standard techniques in this work.

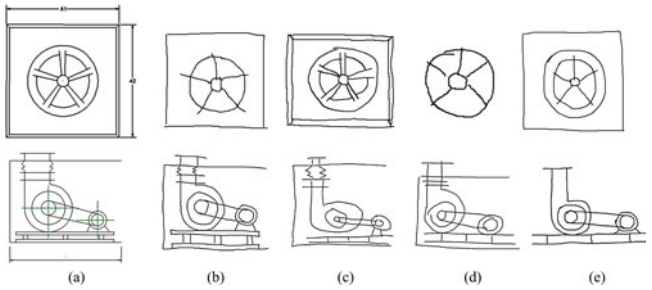

Fig. 1. Examples of large irregularity and variations in hand drawn sketches. (a) Two regular shape templates; (b)-(e) sketches from four different users. See Section 4 for more details on our dataset.

Sketch representation is a fundamental problem and has been extensively investigated. Usually, the representation is based on *geometry* features and *topology* relations of sketch primitives. Previous works vary a lot for such low level features and we refer the readers to a recent survey [1] for more details. Here we briefly review the related work using topology relations. Some approaches [4], [14] only use coarse topology relations like "adjacency" and "inclusion" and focus more on the usage of geometry features. Xu et al. [7] defines five topology relations in more details. Liang et al. [8], [9] further extends the topology relation to eight types, but uses them independently. Almost all previous works [4], [7], [8], [9], [10], [15], [16], [17], [18] use graph representation because it naturally integrates all relational information in a structural manner. Our work differs from previous ones in that we precisely define the eight topology relations in [8], [9] and integrate all of them in a single graph, which we call *fine-grained topology graph*.

Graph matching is a difficult problem. Because sketch retrieval/recognition systems often involve large datasets and require instant feedback, exact matching methods with high complexity are often unaffordable. Previous methods are usually limited to much simplified and inaccurate methods, which fall into two categories. The first category is based on vector reduction, that is, the graph structure is reduced to a numeric vector through eigenvalue analysis, such as Adjacency spectrum (AS) [4], [8], [10] and Laplacian spectrum (LS) [9], [11]. The vector distance is then used as graph similarity. Such methods have constant time computation complexity with respect to the graph and are very fast. However, the vector reduction process does not fully keep the structural information in the graph and results in degraded matching accuracy. Moreover, the vector reduction process is global and it is hard to extract subgraph structures of a shape from its vector representation, making such methods hard for partial matching. The second category adopts approximate graph matching methods, which are usually heuristic and sub-optimal. For example, Xu et al. [7] develop a greedy partial permutation method under certain heuristic constraints. Liang et al. [2] measure the graph similarity by finding the minimum supergraph of the two graphs with a heuristic minimal spanning tree method (MST). In this work, we define the graph similarity as the amount of the maximum common parts between two graphs. It is measured by finding the maximum clique on a novel *topology product graph*. It is intuitive and robust to partial matching. Its computation is optimal and reasonably fast.

Partial matching. In the strict sense, a partial matching algorithm should be able to match any corresponding subparts between two sketches. Because the search space has an exponential complexity in term of the sketch size, the problem is NP-complete and very few previous methods can directly support partial matching. Up to our best knowledge, the only exception is the method in [7], which uses a set of heuristic constraints to prune the exponential search space. However, such constraints are human defined, suboptimal and hard to generalize because they are designed according to the specific task in [7]. Some previous methods [4], [11], [19] support partial matching indirectly. Typically, they decompose each shape in the database into multiple subparts, compute the representation of these subparts in advance, and match a partial query sketch to these subparts independently. In other words, they still perform whole sketch matching and transfers the complexity of partial matching into pre-decomposition, at the cost of largely increased database size and matching time. To deal with the exponential complexity, their sketch decomposition techniques usually exploit task specific characteristics. While such customized techniques work well for the specific tasks, they are hard to generalize to others. For example, the method in [4] only uses "adjacency" and "inclusion" topology relations and designs heuristic sketch decomposition techniques for such relations. The method in [19] only deals with CAD shapes that are closed polygons. By contrary, our method supports partial matching by nature as it encodes all the subpart correspondence information inside the product graph representation. It is more general in that it is not limited to the specific tasks and topology relations that are used, as long as a reasonable product graph representation can be defined.

3 Our Approach

Our approach is described in four stages. The noisy hand-drawn sketches are first refined and decomposed into meaningful shape primitives such as lines and arcs. Such preprocessing has been well studied in the literature. It is briefly described in Section 3.1.

To fully exploit the topology information, we present a comprehensive and precise definition of topology relations between sketch primitives. A *topology graph* is then proposed to integrate all topology information and low level geometry features together, as described in Section 3.2.

To compare two topology graphs, we propose a *topology product graph* that combines the two topology graphs in an exhaustive manner, that is, all the possible common structure correspondences are retained, without any information loss. While the product graph is a general concept in graph theory, this work is the first to exploit the concept in sketch matching problem. To adapt the concept for our problem, we propose effective techniques including a new product operation and edge construction methods. These are detailed in Section 3.3.

We then match two sketches by finding their largest common structure, which is the maximum clique in the topology product graph. We propose a new similarity measure that considers both topology and geometry similarities. Such measure is intuitive and can be easily computed. The details are in Section 3.4.

Overall, our approach improves the state-of-the-art from two aspects. First, the comprehensive topology relations can capture rich semantics in the sketch structure. Second, the new graph representations (topology graph and topology product graph) retain full structural information and motivate an intuitive and accurate similarity measure.

A nice side effect is that, as the product graph and its cliques capture common subgraph structures of the input sketches, our approach can naturally support partial matching. This is important for a practical sketch interface.

3.1 Sketch Primitive Extraction

The raw sketch strokes are noisy and inaccurate. They are first refined and then decomposed into shape primitives before subsequent processing.

In refinement, accidental noises are removed and a clean sketch is produced. We adopt the standard technique in [12]. It has four steps: polygonal approximation, agglomerate points filtering, endpoints refinement and convex hull calculation. An example is given in Fig. 2, where the inaccurate open ended triangle becomes closed and more accurate after endpoint processing. We refer the readers to [12] for more details.

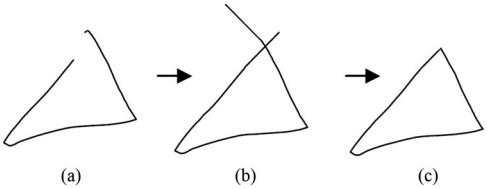

Fig. 2. Examples of endpoint refinement in [12]. (a) a raw sketch; (b) pulling endpoints; (c) deleting extra points.

The refined strokes are then segmented into sub-strokes and fitted to a few primitive shapes (lines, arcs and ellipses) [13] based on pen speed and curvature information. Such primitives serve as a useful low level abstraction of user's input, since there exists a great deal of geometrical variety in different drawing styles and it is difficult to perform sketch matching directly on the strokes.

We note that sketch preprocessing and primitive extraction will affect the overall performance and is a challenging task on its own. After years of investigation, achieving perfect preprocessing is still hard due to the large variety in casual hand sketching. In this work we simply adopt mature techniques that have been used in many previous works and shown working reasonably well in practice. How to improve such preprocessing is a separate research topic and out of the scope of this work.

3.2 Fine-Grained Topology Graph

A sketch is represented as a set of shape primitives. The primitive types are $\{T_{line}, T_{arc}, T_{ellipse}\}$. To extract comprehensive topology and structure information, we define eight topology relations between two primitives, denoted as $\Sigma_R = \{R_{cr}, R_{hc}, R_{ad}, R_{pa}, R_{cu}, R_{ta}, R_{em}, R_{ee}\}$. Such relations are defined according to whether two primitives are intersecting, where the intersection happens and how many intersection points there are. The precise definition is given in Table 1. We note that our definition enumerates all possible cases, as shown in Fig. 3. It is therefore rich and complete. Based on that, we propose a *topology graph* representation to integrate both topology and geometry information as follows.

Definition 1 (Topology Graph). *A topology graph is a four-element tuple* $G = (V, E, \tau, w)$.

TABLE 1
Definition of Topology Relations between Two Primitives P_1 and P_2

Notation	Topology Relation	Definition
R_{cr}	Cross	P_1 and P_2 intersect at a single common inner point.
R_{hc}	Half-Cross	One endpoint of P_1 joins some inner point of P_2.
R_{ad}	Adjacency	1) $P_1^T, P_2^T \in \{T_{line}, T_{arc}\}$; 2) they have a common endpoint.
R_{pa}	Parallelism	1) $P_1^T, P_2^T \in \{T_{line}, T_{arc}\}$; 2) they remain approximately the same distance along their entire length.
R_{cu}	Cut	1) P_1 and P_2 intersect at more than one common inner points; 2) at least one of them is not ellipse.
R_{ta}	Tangency	P_1 and P_2 touch or meet each other at just one common point.
R_{em}	Embody	1) $P_1^T = T_{ellipse}$; 2) P_2 is inside P_1 with no common points.
R_{ee}	EE Intersection	1) $P_1^T = P_2^T = T_{ellipse}$, 2) they intersect at two or more common points.

See Fig. 3 for examples.

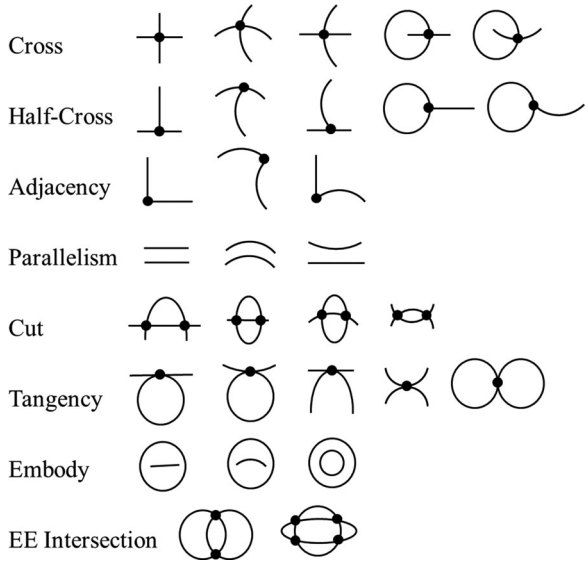

Fig. 3. Examples of topology relations between primitives.

i) *vertices V: each vertex is a primitive in the sketch.*
ii) *edges $E \subseteq V \times V$: each edge (v_i, v_j) is a pair of vertices that has a certain topology relation between them.*
iii) *topology feature function $\tau : E \to \Sigma_R$: $\tau(v_i, v_j) = l$ means that the edge (v_i, v_j) has topology relation l.*
iv) *geometry feature function $w : E \to [0, 1]$: for an edge $e = (v_i, v_j)$, its geometry feature is taken as the spatial distance between the two primitives, and normalized by the whole scale of the sketch so it is scale invariant,*

$$w(e) = \frac{\|c_i - c_j\|_2}{sketch\ scale},$$

where c_i, c_j denote the centroid of primitives v_i, v_j and the sketch scale is the length of the diagonal of the minimal enclosing rectangle of the sketch.

Note that other geometry features between primitives, such as relative orientation, can be used as well, without loss of generality. This is left as future work. We note that the term 'geometry feature' here only refers to the geometrical relation between primitives. The geometry information of individual primitives (length, curvature, ...) has already been used during the preprocessing of primitives.

While many other works [4], [7], [8], [9], [14] also use topology relations and graph representation, we call our method *fine-grained topology graph* because our definition is finer and more precise than previous works, so far the most comprehensive topology definition in the literature of sketch matching. For example, the "Adjacency" relation in [4], [14] is decomposed into seven topology relations as $R_{cr}, R_{hc}, R_{ad}, R_{pa}, R_{cu}, R_{ta}, R_{ee}$ in this work. Our definition is more precise than [7], [8], [9] in that we enumerate all possible structural layouts and define the relations according to the type of the involved primitives and the number of their intersections, which is not done before.

Consequently, our topology graph retains rich topology (and geometry) information of the sketch and better captures the semantic intention of the user. It is discriminative and informative for sketch matching. An example topology graph is shown in Fig. 4.

3.3 Topology Product Graph

As reviewed in Section 2, previous sketch similarity metrics either suffer from structural information loss (vector reduction based methods) or sub-optimality (heuristic graph matching methods).

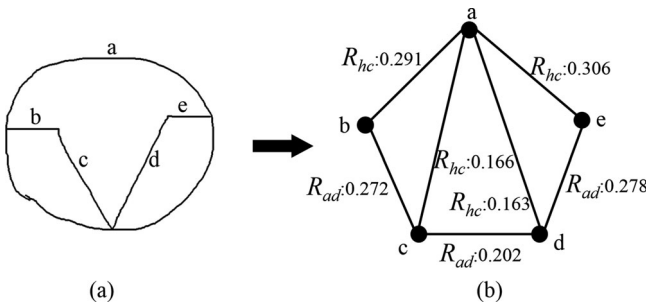

Fig. 4. Fine-grained topology graph. (a) a sketch; (b) its topology graph, shown with topology and geometry features.

We propose a *topology product graph* that fully retains the information for matching two topology graphs. It overcomes the above problems and is much more robust for partial matching. While the concept of product graph is known in graph theory, for the first time we exploit it in sketch matching and propose effective adaptations for the problem on our hand, such as defining a new product operation, how to create edges, and finally a similarity metric that is intuitive and can be exactly computed.

We first define a novel product operation, which we call *Modified Kronecker Product*.

Definition 2 (Modified Kronecker Product). *For two real matrices $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{r\times s}$, the Modified Kronecker Product $A \bar{\otimes} B \in \mathbb{R}^{mr\times ns}$ is a block matrix with m row partitions and n column partitions. It is defined as*

$$A \bar{\otimes} B = \begin{bmatrix} (A \bar{\otimes} B)_{11} & \cdots & (A \bar{\otimes} B)_{1n} \\ \vdots & \ddots & \vdots \\ (A \bar{\otimes} B)_{m1} & \cdots & (A \bar{\otimes} B)_{mn} \end{bmatrix},$$

with each submatrix given by

$$(A \bar{\otimes} B)_{ij} = abs([A_{ij}]_{r\times s} - B) \circ sgn(A_{ij}B),$$

where $[A_{ij}]_{r\times s}$ denotes a $r \times s$ real matrix and all the elements of that matrix is A_{ij}, and $abs(A)$ is a $r \times s$ matrix and each element of $abs(A)$ is the absolute value of the corresponding elements of matrix A; $\circ$ is the Hadamard product [20] that for two matrices of the same dimensions $A, B \in \mathbb{R}^{m\times n}$, $A \circ B \in \mathbb{R}^{m\times n}$, with element given by

$$(A \circ B)_{i,j} = (A)_{i,j} \cdot (B)_{i,j}$$

$sgn(A)$ is the sign matrix of matrix A so that

$$[sgn(A)]_{ij} = \begin{cases} 1, & \text{if } A_{ij} \neq 0 \\ 0, & \text{o.w.} \end{cases}$$

The Modified Kronecker Product operates on two matrices of arbitrary sizes and results in a block matrix. The first term $abs([A_{ij}]_{r\times s} - B)$ in the definition actually computes the absolute difference, i.e. the dissimilarity, of every element pair. While the second term $sgn(A_{ij}B)$ is 1 only when the two elements are both non-zero. Intuitively, it measures the element-wise dissimilarity of two non-zero elements, each from one of the matrices.

We then define the topology product graph as below.

Definition 3 (Topology Product Graph). *Given two topology graphs G and H, their topology product graph $G \times H$ is defined as*

i) *its vertex set is the Cartesian product of the two input vertex sets*

$$V_{G\times H} = V(G) \times V(H).$$

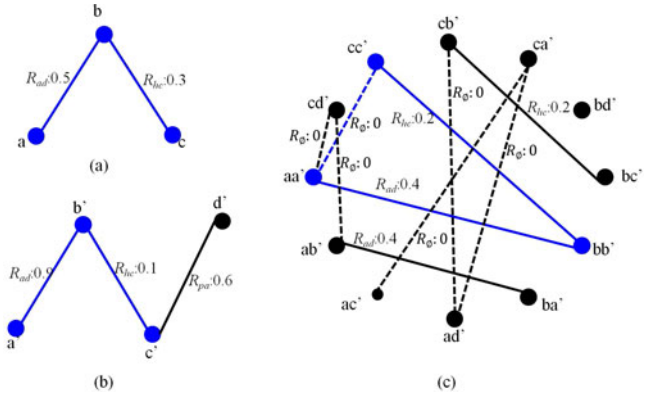

Fig. 5. Construction of topology product graph. (a)(b) two topology graphs; (c) their topology product graph. The common structures are shown in blue. The solid edges in (c) denote the corresponding nodes with the same topology relation (identically labeled edges). The dashed edges denote that no topology relation is presented (no connected edges). The edge weight measures the dissimilarity of the geometry features on two corresponding edges.

ii) *$((u, u'), (v, v'))$ is an edge in $G \times H$ if and only if*

$$\begin{aligned} &((u, u'), (v, v')) \in E_{G\times H} \\ &\quad \Leftrightarrow ((u, v) \in E_G \wedge (u', v') \in E_H \wedge \tau(u, v) = \tau(u', v')) \\ &\quad \vee ((u, v) \notin E_G \wedge (u', v') \notin E_H). \end{aligned}$$

iii) *correspondingly, the topology relation of edge $((u, u'), (v, v'))$ is*

$$\tau((u, u'), (v, v')) = \begin{cases} \tau(u, v), & \text{if } (u, v) \in E_G, \\ R_\emptyset, & \text{otherwise,} \end{cases}$$

where $R_\emptyset$ indicates that no topology relation exists between the corresponding vertices in both input topology graphs.

iv) *the weight matrix of $G \times H$ is the Modified Kronecker Product of the input geometry weight matrices W_G and W_H as in Definition 2*

$$W_{G\times H} = W_G \bar{\otimes} W_H.$$

The topology product graph has an intuitive interpretation. It consists of pairs of identically labeled edges from the two graphs. Each edge in $G \times H$ indicates a common structure shared in G and H: the corresponding vertices in the two graphs either have the same topology relations, or present no relation. The latter case is important as well to measure topology similarity between two sketches.

The edge weight is constructed from Modified Kronecker Product and essentially measures the dissimilarity of the weights between the corresponding edges from the original input graphs. Since we use geometry features as edge weight in original graphs, it measures the geometrical difference of the corresponding primitive pairs with the same topology relation.

We note that the standard Kronecker Product is element-wise multiplication of two matrices. It is usually used to create the adjacency matrix of a product graph. It can find the corresponding edges between the two input graphs, but does not measure the difference between corresponding edge pairs. That is why we call our Definition 2 *Modified Kronecker Product*. One example of the product topology graph is shown in Fig. 5.

3.4 Sketch Matching

A nice property of the topology product graph is that it encodes all the information for matching as its edges enumerates all possible corresponding structures in the two sketches. Naturally, this

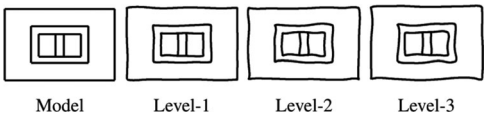

Fig. 6. An example of sketch synthesis from regular shape in [23]. Left: the regular shape. Middle to right: synthesized sketches by three levels of deformation.

means that the maximum clique in the topology product graph is corresponding to the maximum common part with exactly the same topology structures of the sketches. For the example in Fig. 5, the maximum clique (shown in blue) is formed by (aa', bb', cc') and corresponds to the matched edges $((a,b),(a',b'))$, $((b,c),(b',c'))$ and the matched 'no-edge-structure' $((a,c),(a',c'))$, in the two original graphs. This motivates us to define the similarity of two sketches on the maximum clique in their topology product graph, as below.

Definition 4 (Sketch Similarity on Topology Product Graph). *Let the maximum clique of the topology product graph $G \times H$ be a set of edges $E_c = \{e_1, e_2, \ldots, e_{|E_c|}\}$, and the corresponding edge weights be $\{w_1, w_2, \ldots, w_{|E_c|}\}$, the sketch similarity is*

$$sim(G,H) = \alpha \cdot \frac{|E_c|}{max(|E_G|,|E_H|)} + (1-\alpha)\left(1 - \frac{\sum_{i=1}^{|E_c|} w_i}{|E_c|}\right).$$

The similarity measure considers both topology and geometry information. The first term states that a larger clique corresponds to more matched structures and a higher similarity. The second term measures the geometric similarity of the matched structures, noting that the edge weight in the topology product graph is the difference of edge weights from the input graphs, as described in Definitions 2 and 3. Both terms are normalized into $[0,1]$ and balanced by the parameter α, which is empirically set to 0.5 in our experiments so the two terms are equally important.

The proposed similarity measure naturally supports partial matching. Because the topology product graph retains all matched structures, an incomplete input sketch would still form a matched part and this whole structure can be found as a clique.

Now the problem is to find the maximum clique. In general, it is NP-hard and has been well studied in graph theory. The algorithm with best theoretical complexity is by Robson [21]. It has complexity $O(1.1888^n)$ with graph size n. In practice, various heuristic yet exact algorithms have been proposed. Even though their worst case complexities are exponential, they are shown working well in practice and run reasonably fast for small graphs. One category of such methods uses branch-and-bound to reduce the search space. In our implementation, we adopt the best performing algorithm in this category, the MaxCliqueDyn algorithm [22]. Using a simple $O(n^2)$ sorting function in the recursion, it is much faster than its competitors. We refer the readers to [22] for more details.

4 Experiments

A new challenging dataset. There are few standard common datasets for sketch recognition and retrieval. This is partially because sketch based applications are usually domain specific, e.g. garment design [2], and partially because collecting a sufficiently comprehensive and diverse hand drawn dataset takes a lot of efforts.

Most sketch recognition works are based on synthesized data from graphic symbols through certain deformation [23]. They are relatively simple in that the shapes are usually regular and consist of only a small number of strokes. To simulate the irregularity in human sketching, the regular shapes are slightly deformed, as exemplified in Fig. 6. Such synthesized sketches are not as challenging as real ones. As a result, previous methods usually work pretty well on such datasets.

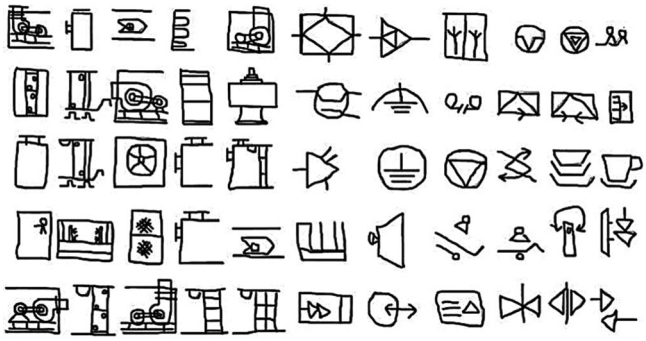

Fig. 7. Examples of hand drawn sketches in our new dataset.

To reflect the real challenges in hand drawn sketch matching, in this work we create a dataset with large diversity and magnitude. It consists of 55 template shape classes, with 30 from engineering unit prototypes and 25 from electric circuit diagrams. We asked 10 users to follow the templates and draw two sketches for each class. Totally 1,100 sketches are collected. After discarding 14 bad drawings, we created a dataset with 1,086 sketches. As illustrated in Figs. 1 and 7, the drawn shapes are quite irregular and the variations within the same classes are extraordinarily large, showing that users have different drawing styles and perceptions about sketching. This dataset is therefore more rigorous and appropriate to test the performance of sketch matching methods.

Comparison to the state-of-the-art. On the challenging dataset, our proposed approach is compared to several state-of-the-art sketch matching methods that also use graph based representation, including the minimal spanning tree method [2], Laplacian spectrum with [9] and without geometry matching [9], [11], and Adjacency spectrum matching [4], [10]. As discussed in Section 2, these methods suffer from either loss of structural information or inaccuracy of approximate graph matching.

We note that in the original literature some methods in comparison use different (typically simpler) topology or geometry features from ours and therefore cannot be directly compared. For fair comparison we adapt these methods to use the same eight topology relations in Section 3.2. Therefore, their performance difference to ours is attributed to our new topology graph construction and product graph based similarity metric. Specifically, MST [2] only deals with closed shapes and uses task-specific geometry features in graph construction. However, the MST matching metric is quite general and can be used to match our topology graphs, allowing a fair comparison with our matching method. For LS_G and LS, we follow the same experiment settings in [9] to use the same eight topology relations in this work. We note that LS_G is an enhanced version of LS. It uses rich geometry features that are more complex than ours. The original AS [4] only uses two coarse topology relations ("adjacency" and "inclusion") and similar geometry features as ours. It performs poorly in our experiment. For fair comparison, we enhance the AS method by using the same eight topology relations in the same way of their original graph construction. The enhanced version, called AS$*$, is much better and is used in our comparison. For completeness we show the results for both AS and AS$*$.

Overall comparison. We perform a series of sketch retrieval trials, 10 times for each class. In each trial, the query sketch is randomly selected. To test partial matching, the query sketch is made incomplete by randomly removing a subset of its primitives. Let the amount of incompleteness be a ratio $\mu \in [0,1]$, for complete matching we use $\mu = 0$ and for partial matching we use $\mu = 1/3$, which means one third of the query sketch is removed.

The retrieved sketches are sorted in a descending order of their similarities to the query. A retrieved result is considered correct if it belongs to the same class as the query, and wrong otherwise. We obtain precision/recall values and use precision-recall curve

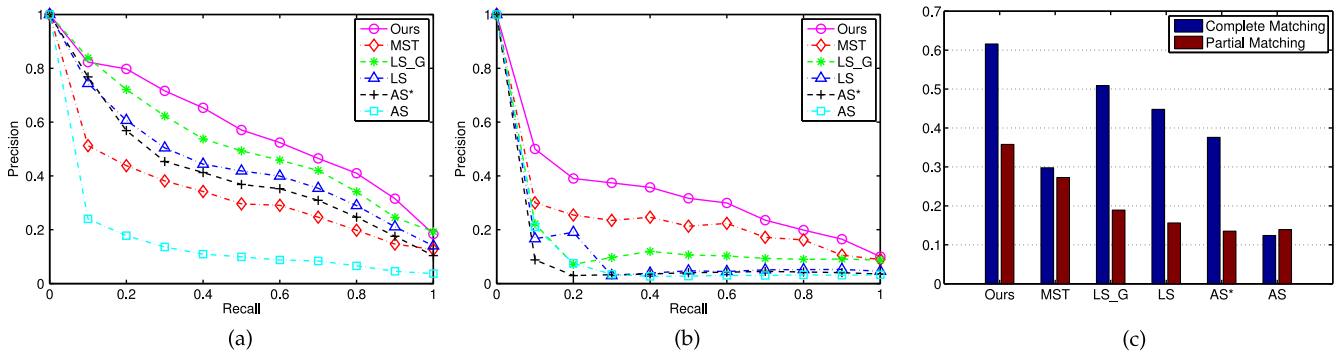

Fig. 8. Comparison of six methods for complete and partial matching. (a) PR curves for complete matching; (b) PR curves for partial matching; (c) Mean Average Precisions for complete and partial matching.

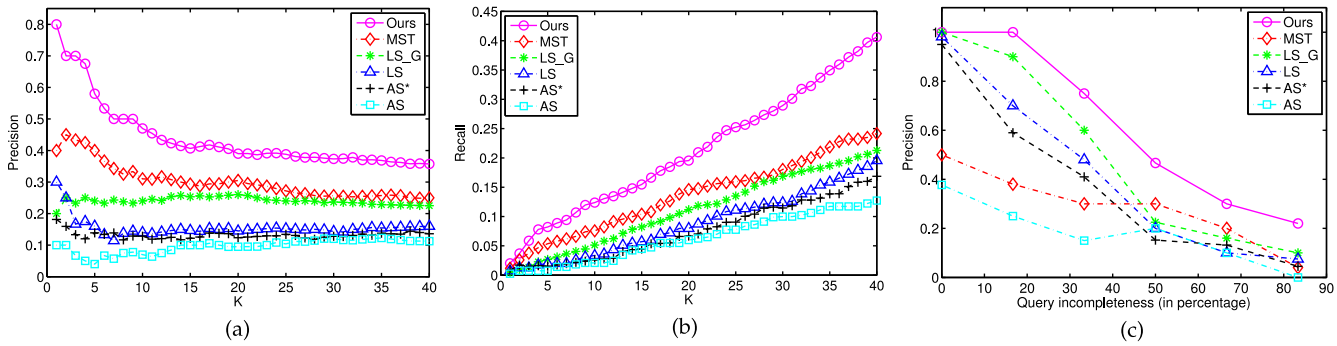

Fig. 9. Comparison of six methods for partial matching. (a)(b) Average precision/recall for different Ks; (c) Average precision with respect to the incompleteness of the query shape for $K = 1$.

(PR curve) as performance metric. This is standard for evaluation of a retrieval system [24]. Note that the PR curves are averaged over all classes.

The PR curves and the Mean Average Precisions of the six approaches in comparison are depicted in Fig. 8, for both complete and partial matching. Not surprisingly, the accuracy of all methods drops from complete to partial matching. The results show that our approach significantly outperforms other methods. Especially for partial matching, our precision is in general 50 percent higher than the second-best method, and far better than the others.

There is an interesting observation. In partial matching, the runner up is another graph matching method (MST), which is based on heuristics and approximate rules. The fact that the best two methods are based on graph matching indicates that preserving the structural information and matching subgraphs is important for a partial query. The remaining three methods reduce the whole graph into a single numeric vector. Such a holistic representation is sensitive to small and local variations, as only changing a local part could change the entire representation. Thus they are poor for partial matching.

However, in complete matching the MST method is beaten by the best vector reduction based methods (LS_G, LS and AS$*$). We believe this is due to its heuristic and non-optimal graph matching that results in a lot of false positives. This further consolidates the importance of both good graph representation and matching method, which are the main contributions of this work.

More results in partial matching. In practice, sometimes the accuracy of the top K retrieved results is used to evaluate the performance. We show the precisions and recalls of partial matching ($\mu = 1/3$) with a varying K in Figs. 9a and 9b. Our method consistently outperforms others over all different Ks. Especially, if we only consider the first result ($K = 1$), our precision is very high (80 percent) and almost twice better than the second best method. We further investigate 6 different levels of incompleteness at $\mu = l/6, l = 0, 1, 2, 3, 4, 5$. Fig. 9c shows the precision of all methods when $K = 1$. While all methods become worse as μ increases, our method is much more accurate. It still achieves 50 percent precision even when half of the query is removed ($\mu = 0.5$).

Fig. 10 shows example retrieval results with all methods. Each row shows the partial query with a shadow background at left,

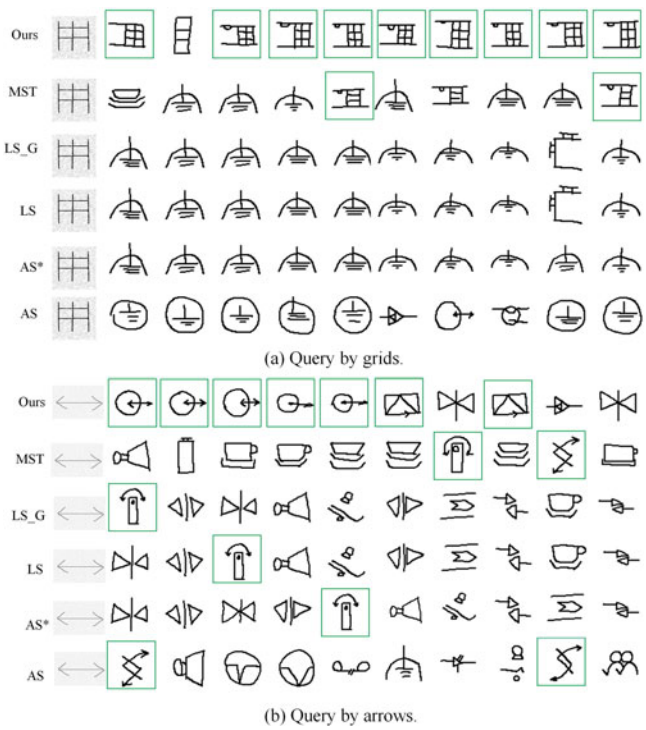

Fig. 10. Example partial matching results using six different methods. The leftmost sketch in the shadow is the partial query. For each query the top 10 matched sketch shapes (from left to right) from the database are shown. The positive results are labeled with green frames.

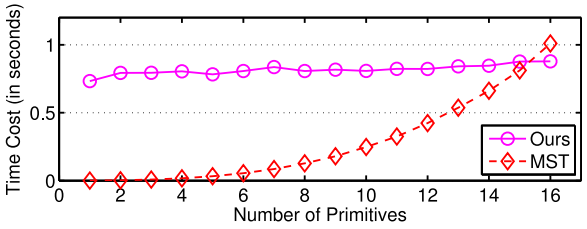

Fig. 11. Runtime of two graph based matching methods with increasing number of primitives in the query sketch.

followed by the top 10 retrieval results. Our method can often correctly match the partial shape, while the results returned by other methods have many errors. Using our method, users can find their intended sketch shapes by drawing the query in a coarse-to-fine manner, often without completing the query.

Runtime analysis. All methods are implemented in C++. Runtime is measured on an Intel Core(TM)2 2.4 GHz CPU. For the vector reduction based methods (LS_G, LS, AS$*$ and AS), the vector representations of all sketches in the dataset are computed offline and their computation is simply vector comparison, which is very fast and irrelevant to the complexity of the query. Searching the entire dataset takes about 0.1 second (0.153 for LS_G, 0.129 for LS, 0.122 for AS$*$ and 0.099 for AS).

Our approach and MST method are slower because they perform graph matching for each sketch in the dataset, and their runtime increases with respect to the complexity the query. Fig. 11 reports their average runtime with respect to the number of primitives in the query. Our maximum clique finding algorithm is slower, but its computational complexity moderately increases with the graph size. The heuristical MST method is in general faster, but its complexity increases quickly with the graph size and does not scale well with large graphs.

Discussions. By its nature, our approach is accurate but relatively slow. This makes it hard to directly apply our approach for large scale database search with complex shapes such as diagrams. We note that targeting both high accuracy and speed is probably too challenging for any single sketch matching method. Such goals are usually realized through heavy engineering efforts such as careful integration of different methodologies and tuning of various speed-accuracy trade-off. Our approach could be a useful ingredient in such engineered systems, with proper adaptation. First, for highly complex shapes our approach can be made more efficient with a hierarchical representation, where each hierarchy level adopts different topology relations and graphs, probably in a specified coarse-to-fine manner. Comparison can be performed in a cascade manner more efficiently: a finer level graph is compared only if the matching scores on all previous coarse levels are good enough. Second, for large scale databases, faster but less accurate methods (e.g. the vector reduction based methods) can be first used to retrieve a number of candidate results. Such results are then re-ranked using our method for better accuracy. This is a common practice in modern search engines.

5 Conclusions and Future Work

It is the first time that the concept of product graph is exploited for sketch matching. It turns out quite effective, combined with the comprehensive definition of topology relations, effective graph construction methods and an intuitive similarity metric.

Although its inherent computational complexity is exponential, in practice the speed of our method is reasonably fast (see Fig. 11). Furthermore it can be easily combined with previous faster but less accurate methods for better performance.

Our current method is deterministic. Therefore, it is hard to correct the errors from the preprocessing and topology relation classification steps. One possible future work is to adopt probabilistic representations for sketch primitives, topology relations and the product graph to define a more robust similarity metric.

Acknowledgments

This work was supported by The National Science Foundation of China (No. 61305091, No. 61305094, No. 11271351), The Fundamental Research Funds for the Central Universities (No. 2100219038), and Shanghai Pujiang Program (No. 13PJ1408200), Shuang Liang is the corresponding author.

References

[1] T. Lu and L. Wenyin, "Sketching interfaces," in *Handbook of Document Image Processing and Recognition*, D. Doermann and K. Tombre, Eds. New York, NY, USA: Springer-Verlag, 2014.
[2] S. Liang, R. H. Li, and G. Baciu, "A graph modeling and matching method for sketch-based garment panel design," in *Proc. 10th IEEE Int. Conf. Cogn. Inf. Cogn. Comput.*, 2011, pp. 340–347.
[3] W. Y. Liu, X. Kong, Y. Wang, C. Wan, C. Y. Ho, T. Lu, and Z. Sun, "QuickDiagram: A system for online sketching and understanding of diagrams," in *Graphics Recognition: Achievements, Challenges, and Evolution*, vol. 6020, J. M. Ogier, W. Liu, and J. Lladós, Eds. New York, NY, USA: Springer, 2010, pp. 130–141.
[4] M. Fonseca, A. Ferreira, and J. Jorge, "Sketch-based retrieval of vector drawings," in *Sketch-based Interfaces and Modeling*, J. Jorge and F. Samavati, Eds. New York, NY, USA: Springer, 2011, pp. 181–201.
[5] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa, "Sketch-based Shape Retrieval," *ACM Trans. Graph*, vol. 31, no. 4, pp. 31:1–31:10, 2012.
[6] X. Sun, C. Wang, C. Xu, and L. Zhang, "Indexing billions of images for sketch-based retrieval," in *Proc. 21st ACM Int. Conf. Multimedia*, 2013, pp. 233–242.
[7] X. Xu, Z. Sun, B. Peng, X. Jin, and W. Liu, "An online composite graphics recognition approach based on matching of spatial relation graphs," *Int. J. Document Anal. Recognit.*, vol. 7, no. 1, pp. 44–55, Mar. 2004.
[8] S. Liang, Z. Sun, and B. Li, "Sketch retrieval based on spatial relations," in *Proc. Int. Conf. Comput. Graph., Imag. Vis.*, 2005, pp. 24–29.
[9] S. Liang and Z. Sun, "Sketch retrieval and relevance feedback with biased SVM classification," *Pattern Recognit. Lett.*, vol. 29, no. 12, pp. 1733–1741, Sep. 2008.
[10] P. Sousa and M. J. Fonseca, "Sketch-based retrieval of drawings using spatial proximity," *J. Vis. Lang. Comput.*, vol. 21, no. 2, pp. 69–80, 2010.
[11] M. F. Demirci, R. H. van Leuken, and R. C. Veltkamp, "Indexing through Laplacian spectra," *Comput. Vis. Image Understanding*, vol. 110, no. 3, pp. 312–325, Jun. 2008.
[12] J. Xiangyu, L. Wenyin, S. Jianyong, and Z. Sun, "On-line graphics recognition," in *Proc. 10th Pacif. Conf. Comput. Graph. Appl.*, 2002, pp. 256–264.
[13] T. Sezgin, "Feature point detection and curve approximation for early processing of freehand sketches," Master's thesis, Dept. of Electr. Eng. Comput. Science, MIT, Cambridge, MA, USA, 2001.
[14] W. H. Leung and T. Chen, "Retrieval of sketches based on spatial relation between strokes," in *Proc. Int. Conf. Image Process.*, 2002, vol. 1, pp. 908–911.
[15] M. Pelillo, K. Siddiqi, and S. Zucker, "Matching hierarchical structures using association graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 11, pp. 1105–1120, Nov. 1999.
[16] C. Li, B. Yang, and W. Xie, "Online hand-sketched graphics recognition based on attributed relational graph matching," in *Proc. 3rd World Congr. Intell. Control Autom.*, 2000, vol. 4, pp. 2549–2553.
[17] J. Lladós, E. Martí, and J. J. Villanueva, "Symbol recognition by error-tolerant subgraph matching between region adjacency graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 10, pp. 1137–1143, Oct. 2001.
[18] R. H. van Leuken, O. Symonova, and R. C. Veltkamp, "Topological and directional logo layout indexing using hermitian spectra," in *Proc. 5th IASTED Int. Conf. Signal Process., Pattern Recognit. Appl.*, 2008, pp. 93–98.
[19] S. Berchtold and H.P. Kriegel, "S3: Similarity search in cad database systems," in *Proc. Int. Conf. Manage. Data*, 1997, pp. 564–567.
[20] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1990.
[21] J. Robson, "Finding a maximum independent set in time $o(2^{n/4})$," LaBRI, Universit Bordeaux I, Bordeaux, France, Tech. Rep. 1251-01, 2001.
[22] J. Konc and D. Janezic, "An improved branch and bound algorithm for the maximum clique problem," *MATCH Commun. Math. Comput. Chem.*, vol. 58, pp. 569–590, 2007.
[23] M. Luqman, M. Delalandre, T. Brouard, J.Y. Ramel, and J. Lladós, "Fuzzy intervals for designing structural signature: An application to graphic symbol recognition," in *Graphics Recognition Achievements, Challenges, and Evolution*, vol. 6020, J. M. Ogier, W. Liu, and J. Lladós, Eds. Berlin, Germany: Springer, pp. 12–24, 2010.
[24] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.