Characterizing the Performance of an Image-Based Recognizer for Planar Mechanical Linkages in Textbook Graphics and Hand-Drawn Sketches

Matthew Eicholtz, Levent Burak Kara*

Department of Mechanical Engineering, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Abstract

In this work, we present a computational framework for automatically generating kinematic models of planar mechanical linkages from raw images. The hallmark of our approach is a novel combination of supervised learning methods for detecting mechanical parts (e.g. joints, rigid bodies) with the optimizing power of a multiobjective evolutionary algorithm, which concurrently maximizes image consistency and mechanical feasibility. A rigorous set of experiments was conducted to systematically evaluate the performance of each phase in our framework, comparing various combinations of joint and body detection schemes and feasibility constraints. Precision-recall curves are used to assess object detection performance. For the optimization, in addition to standard accuracy measures such as top-N accuracy, we introduce a new performance metric called *user effort ratio* that quantifies the amount of user interaction required to correct an inaccurate optimization solution. Current state-of-the-art performance is achieved with (i) one (or a cascade of) support vector machines for joint detection, (ii) foreground extraction to reduce false positives, (iii) supervised body detection using normalized geodesic time, distance, and detected joint confidence, and (iv) feasibility constraints derived from graph theory. The proposed framework generalizes moderately well from textbook graphics to hand-drawn sketches, and *user effort ratio* results demonstrate the potential power of an interactive system in which simple user interactions complement computer recognition for fast kinematic modeling.

Keywords: computer vision, evolutionary multiobjective optimization, image processing, kinematic modeling, object recognition, sketch recognition

1. Introduction

A planar mechanical linkage is an assembly of rigid bodies connected by kinematic pairs (or joints) that constrain its motion within a plane. With applications in robotics [6, 7], healthcare [29, 38], transportation [58, 65], and industrial equipment [32, 46], among others, we observe and make use of the dynamic behavior of complex mechanical systems on a daily basis. Visualizing the coordination motion of mechanical linkages is indeed a valuable skill for improving design intuition [18], yet during the design and analysis of such dynamic assemblies, the visual content is largely static in nature, as illustrated in Figure 1.

To overcome this information deficit, students and engineers may use mental simulations to infer mechanical behavior [36], but this can be difficult for complex problems [35] or for individuals with low spatial ability [37]. They also frequently use hand-drawn sketches to convey design ideas, perhaps incorporating key annotations and arrows to demonstrate motion; even so, the burden for visualization remains with the user. Alternatively, computer simulations can be generated using specialized software [1–4], in which users manually create kinematic models. However, this task is often too time-consuming

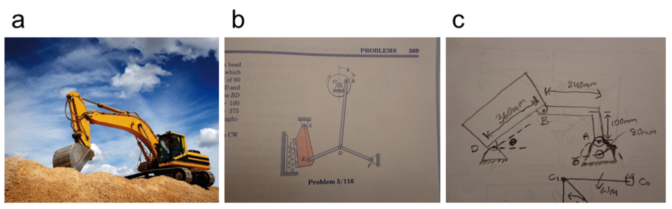

Figure 1: Example mechanical linkages (a) in the real world, (b) in textbooks, and (c) in hand-drawn sketches[2]. The latter two cases represent valid inputs for the recognition framework presented in this paper.

to be practical (e.g. students solving a dynamics homework problem, professional engineers brainstorming potential design concepts) and may require advanced programming skills, hindering novice users. There is a clear need for better software tools that facilitate quick computer-based kinematic visualization of mechanical linkages, filling the gap between ineffective mental simulations and impractical manual model creation.

In this paper, we address that need by proposing a computational framework that automatically recognizes the underlying mechanical structure in images of textbook graphics or hand-

*Corresponding author.
Email addresses: `meicholt@andrew.cmu.edu` (Matthew Eicholtz), `lkara@cmu.edu` (Levent Burak Kara)

[2]Ironically, even the excavator shown here is static in this printed document, despite our usual observation of its dynamic behavior in the real world. Images courtesy of (a) MOBA (www.moba.de), (b) the MECH135 dataset, and (c) the MECHS250 dataset.

drawn sketches. In order to generate a proper kinematic simulation of a planar mechanical linkage, the user typically needs to specify the number and position of rigid bodies that make up the linkage, as well as the type and location of joints that dictate relative motion between rigid bodies. The goal of our method is to offload this burden to the computer. We accomplish this in multiple stages using a joint-centric approach, meaning that linkages are viewed as a collection of connected joints (as opposed to connected bodies), and each pairwise joint connection indicates that those two joints exist on the same rigid body. In this way, rigid bodies can be inferred from the connections between joints. This reduces the problem to localizing joints in the image, predicting which joints coexist on rigid bodies, and resolving discrepancies to form reasonable mechanical assemblies. For simplicity, we limit our study to planar mechanisms comprising only revolute (pin) joints.

2. Background

Our computational approach relies on ideas from many disciplines, including computer vision, sketch recognition, and evolutionary computation. Here, we outline key references in these areas that influenced the design of our recognition framework.

2.1. Scene understanding

The task of identifying structured objects in images is not a new one [14, 24, 25]. Practical applications include face recognition [26, 66], pose estimation [64], and 3D surface estimation [55]. The key difference, though, between previous work in this area and our present domain is that planar mechanisms do not have well-defined structural or spatial dependencies. For example, in face recognition, it is straightforward to learn that a chin should not be located above the nose or that eyes should exist between the ears; with mechanical linkages, it is less clear if a specific joint should be systematically connected to another. Little knowledge is gained about the likelihood of other objects in the image just from knowing one object's location.

Within the current domain of interest, Sato et al. [57] proposed a vision-based approach for automatically estimating the location of an axis of rotation in a mechanical linkage. The primary differences between that work and the research in this paper are twofold. First and foremost, it relies on motion tracking from a series of images to capture the moving parts, whereas our work is restricted to a single image. Second, it seems to be limited to simple mechanisms with only one axis of rotation (single pin joint), whereas our approach can handle more complex kinematic behavior (multiple pin joints).

2.2. Sketch recognition

Two important aspects of sketch recognition that relate to the present work are representation and complexity. With regard to representation, two classes of techniques have emerged in the literature. Stroke-based methods treat each sketch as a sequence of time-stamped strokes, each containing a series of sample points in space. While some works share similarities to our domain [17, 33, 34, 42, 52], stroke-based methods are ill-suited for our recognition framework, which is designed to work on rasterized images. Still, there are interesting parallels; for instance, [34] uses a graph representation to combine low-level primitives into high-level shapes using geometrical rules. We also implement graphs in our recognition pipeline, but instead connect low-level joints to form high-level mechanisms based (partially) on mechanical feasibility rules.

The other class of sketch recognition techniques is image-based approaches, including the present work, which neglect temporal information and only consider the spatial layout of pixels. This poses the additional challenge of grouping relevant pixels, depending on the object being recognized. With regard to sketch complexity, it is important to distinguish between isolated symbol recognizers and detecting objects in freehand sketches, which is a more challenging problem. The task of symbol recognition can be treated as a template matching problem; some examples of successful approaches in this area include [13, 28, 39, 40, 43, 50]. In some sense, the joint recognition algorithm used here is similar to a sliding window symbol recognizer. However, due to the allowable shape variance of objects in mechanical linkages, we do not use unsupervised part templates and instead learn a discriminative model based on local image features.

Within the current domain of interest, researchers have briefly studied the automatic recognition of mechanical systems from sketch input [17, 27, 28], but these approaches typically involve clean images, well-defined part templates, and sometimes make use of temporal information to aid recognition. Our approach must generalize well to raw images, which are often noisy, may contain extraneous information from other graphics, and do not always contain well-defined part models.

2.3. Evolutionary multiobjective optimization

The proposed framework includes an evolutionary optimization stage to resolve discrepancies from the vision-based detection of joints and joint connections. There is a growing body of research in the area of multiobjective evolutionary algorithms (MOEAs), especially in regard to real-world applications. Many well-known MOEAs are based on Pareto dominance [19, 61, 67], which states that a given solution dominates another solution if it is at least as good on all objectives and better on at least one objective. Arguably the most popular MOEA of this type and the one used in this present work is the nondominated sorting genetic algorithm (NSGA-II) [19], which has been successful largely due to fast computation of nondominated fronts, preservation of elitist solutions, and lack of a user-specified sharing parameter.

The nondominated sorting genetic algorithm was first introduced almost 20 years ago (NSGA, [61]) and improved 8 years later (NSGA-II, [19]). The two identifying characteristics of NSGA-II are nondominated sorting and crowding distance. Nondominated sorting involves locating the Pareto front (all nondominated solutions), assigning those solutions a rank of 1, and iteratively assigning higher ranks to each Pareto front level, ignoring all previously detected levels. Crowding distance measures the local spread of solutions and is used to preserve diversity in the population. These parameters are used during the selection process as follows. When sorting a set of solutions, any solution with lower (better) rank goes before solutions with higher rank. Inevitably, there will be cases when the desired number of survivors cuts through one of the Pareto ranks. In this case, the solutions with that rank are sorted with preference given to higher crowding distances. This algorithm has been shown to have success when the number of objectives is small, so it should be a suitable approach for the problem presented in this paper.

A number of researchers have applied evolutionary algorithms to the domain of mechanical linkages [5, 9, 41, 44, 45, 49]. However, this line of work is concerned with synthesis of new mechanisms rather than analysis of existing ones. Generally, the optimization goal is to evolve mechanical linkages (many times with fixed topology) for which a coupler point follows a target trajectory path. Additional constraints on the parameter space make the problem tractable. By contrast, our optimization goal is to evolve mechanical linkages that are consistent with the visual content in an image and kinematically feasible; no path trajectories are prescribed. While these related works on mechanism synthesis are confined to a well-defined parameter space, our approach must deal with the additional challenges of object recognition.

For the present domain, the feasibility of a predicted mechanism is governed by mechanical principles. These principles can be formulated as a series of constraints; in this way, large regions of the search space may become infeasible because one or more of the constraints fail. A critical step in algorithm design is determining how to handle such constraints. Constraint handling methods can be broadly categorized into two groups: (i) those that always prefer feasible solutions (hard constraints) [12, 19] and (ii) those that treat constraints as objectives (soft constraints) [62]. We employ the latter method in order to allow infeasible, yet strong, solutions to persist because they may be near the constraint boundaries.

3. Related work and contributions

The current work builds upon two previous attempts to solve the problem of automatic image-based kinematic modeling of mechanical linkages. In our first effort [22], we introduced a baseline approach comprising a fixed-window sliding object detector to localize probable joints, an unsupervised metric called normalized geodesic time to predict which joints should be connected to each other, and the NSGA-II algorithm [19] to evolve a small set of feasible mechanical structures using the vision output.

Despite initial promising results, there was room for improvement in each phase of the approach. Indeed, one of the benefits of our framework is that core modules (e.g. joint detection, body detection, structural optimization) can be modified in isolation without affecting the underlying principles of the approach. With this in mind, in a secondary work [21], we enhanced the joint detection algorithm by incorporating multiple context-based classifiers of increasing window size and implementing a greedy foreground extraction technique. Broadly speaking, these modifications had the effect of increasing the confidence gap between true positives (actual joints in the image) and false positives (incorrectly-identified joints) as well as descreasing the total number of false positives, respectively. In addition, we investigated whether training our system on images of textbook graphics could transfer well to tests on hand-drawn sketches and vice versa. The primary contribution of that work was the idea that it may be possible to build a powerful sketch recognition tool without ever needing a sample sketch for learning.

In the present work, we shift our focus to the systematic evaluation of each phase in our framework. Previously, quantitative comparisons between algorithm variants were made only at the optimization level. In this paper, we incorporate precision-recall curves to assess the quality of joint and body detection schemes. Also, the performance metrics previously used to evaluate accuracy of evolved solutions were somewhat coarse – that is, a binary indicator of success for a given image may not fully characterize the quality of the optimization routine. Mechanical linkages are complex, with many examples containing ten or more parts. We believe a recognition framework that regularly identifies 90% of the required components successfully, for example, is better than one that only identifies 10% of the required components. To that end, we propose a novel metric called user effort ratio that expresses the amount of user effort required, on average, to correct a solution provided by our optimization routine. In effect, this metric exemplifies the potential benefit of using our system versus fully manual construction of kinematic models.

In addition to the evaluation goals of this work, we also continue to improve our algorithms by integrating a supervised method for learning pairwise joint connections and creating more robust optimization constraints on mechanical feasibility. In summary, the significant contributions of this paper beyond previous related work include:

1. Inclusion of a supervised learning approach for body detection.
2. Development of new feasibility constraints for optimization based on graph theory.
3. Introduction of a new performance metric (user effort ratio) for assessing overall solution quality.
4. Systematic evaluation of each stage in the recognition framework, comparing recent algorithm modifications to all previously reported methods.

4. Overview

The proposed methodology for recognizing mechanical linkages in images is illustrated in Figure 2. A database of images with labeled ground truth information – mechanical structure including joint locations and pairwise connectivity – is required for training. During testing, the input to our algorithm is a single image containing a planar mechanical linkage. Model parameters are unknown *a priori*, including the number of joints, number of bodies, topology, and geometric configuration. Although we initially targeted textbook graphics, the framework can be applied to images of hand-drawn sketches without any

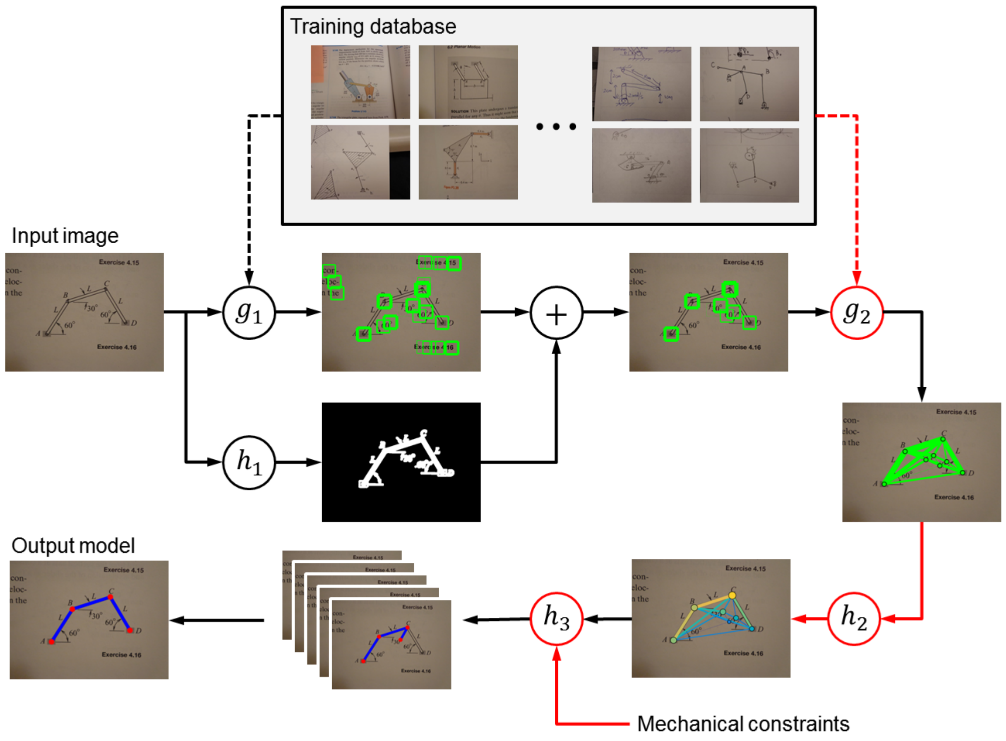

Figure 2: Overview of the proposed recognition framework. Taking a raw image as input, probable joints are detected using a supervised classifier (g_1) and filtered using foreground extraction (h_1). Then, rigid bodies are inferred from the supervised detection of pairwise joint connections (g_2), removing any remaining joints that do not have any connections (h_2). The resulting data is passed to an evolutionary algorithm (h_3) that optimizes a graphical model based on fitness criteria related to image consistency and mechanical constraints.

modifications. No explicit restrictions are made regarding the position, scale, or orientation of the linkage within the image, although it is assumed the entire mechanism is fully visible. In addition, images do not need to be pre-processed in any way (e.g. cropping, filtering), so they may contain noise, illumination changes, and extraneous information such as text, annotations, pencil markings, or partial depictions of other mechanical systems.

The fundamental principle of our approach is to identify salient components that make up mechanical linkages first, and then to optimize the collection of detected components into structural graphs such that results are consistent with visual cues in the image and plausible in terms of engineering mechanics. Mechanical components are detected in a serial manner. Probable joints are identified first (g_1), with an optional filtering step to ignore background candidates (h_1). Then, the likelihood of body connections between joints is estimated (g_2), followed by a simple filter to remove joints without any connections (h_2). The optimization (h_3) takes into account the confidence of all vision-based detections as well as mechanical principles to ensure that feasible structures are preferred. The output is a small set of kinematic models that can be easily integrated into engineering software packages for further analysis and simulation.

In the next section, we discuss the technical details of our algorithms. As stated earlier, portions of the proposed framework were developed in earlier papers [21, 22]. For completeness, we describe the entire framework here, but emphasize the components that are unique to this paper, which are identified in Figure 2 by red lines.

5. Technical details

5.1. Detecting pin joints

The first step in the vision pipeline is to identify the location of joints in an input image. We accomplish this task by running a sliding window over the entire image and classifying each patch using a linear support vector machine (g_1 in Figure 2). The SVM is trained over HOG features [15] computed on a set of example images; this process is outlined in Figure 3.

The training data comprises positive and negative image patches extracted from selected hand-labeled images. Positive samples contain a pin joint, while negative samples do not. The negative patches are randomly selected from the training images, making sure they do not overlap with positive samples. Similar to Fu and Kara [27], we augment the training set by applying a series of simple transformations to each positive example: reflection about the vertical or horizontal axis, and rotation by 90, 180, and 270 degrees. This effectively increases the positive training data by a factor of 5, which likely improves the discriminative power of the joint detector. In addition, we follow the approach of Dalal and Triggs [15] to improve classification accuracy by mining hard negatives after initially training

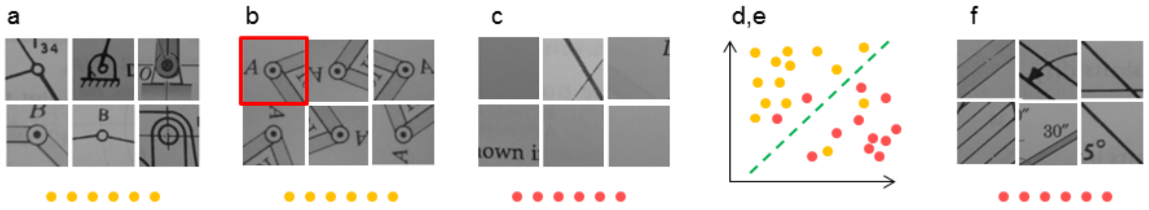

Figure 3: Training a support vector machine for joint detection. (a) Extract positive examples of joints in training images; (b) Augment the dataset by reflection and rotation; (c) Extract random negative examples from images; (d) Compute features for all examples; (e) Train a soft linear support vector machine; (f) Mine hard negatives from training images and retrain the support vector machine.

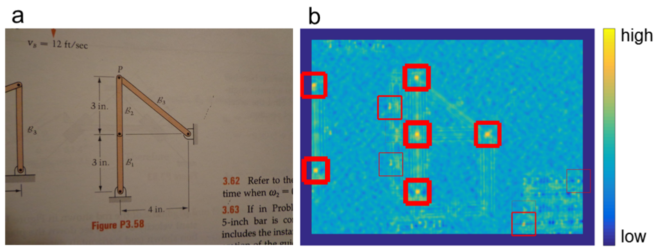

Figure 4: (a) An example image and (b) its corresponding heat map for joint detection, where color encodes distance to the decision boundary and bounding boxes highlight local maxima. Bounding box thickness positively correlates with detection confidence.

the SVM. This is especially critical for our problem domain because most of the background in textbook images and sketches is blank, and therefore the initial set of negative training examples may not accurately reflect the diversity of the negative image space.

After training, the SVM can be used on a test image to classify the image patch centered at each pixel. The result is a heat map, in which distance from the decision boundary (hyperplane) encodes detection confidence. In our method, we ignore pixels with confidence less than zero (negative distance to the hyperplane indicates the patch is more likely to *not* contain a joint) and then apply the non-maxima suppression technique in [24] to isolate local maxima. Figure 4 illustrates this concept on an example image. The remaining detected joint locations and associated confidence values are then passed to later stages of the algorithm.

5.1.1. *Multiple context-based classifiers*

Preliminary experiments using the fixed-window SVM classifier described above revealed that text and annotations in an image frequently trigger false joint detections with high confidence. These false positives can be problematic for our optimization routine, which relies on detection confidence in the computation of multiple fitness objectives (see section 5.3.2). With this in mind, we introduced the idea of using context cues from larger neighborhoods in the image to better discriminate between true and false joints [21]. More specifically, we train three SVM classifiers on HOG features over increasing window size and employ a weighted sum of the distances to each decision boundary as the indicator of detection confidence. The HOG descriptor parameters are modified according to window size such that all image patches have the same number of features; in other words, larger window sizes yield coarser spatial binning. Unlike many multi-scale classification schemes, we do not run all classifiers in parallel on the full image. Instead, we implement a serial approach, in which the root classifier instantiates potential joint detections and then the two context classifiers adjust the confidence level of those detections accordingly. Refer to [21] for additional details regarding classifier weights and example results for this detection scheme.

5.1.2. *Foreground extraction*

The inclusion of multiple context classifiers has the primary benefit of increasing the confidence gap between true positive joints and false positive joints, on average. In order to increase the overall precision of the joint detection scheme, i.e. to reduce the number of false positives, we filter out background detections using a greedy foreground extraction algorithm (h_1 in Figure 2). The algorithm is depicted in Figure 5 and proceeds as follows. First, a Sobel edge detector [20] is run over the saturated grayscale image, yielding a set of boundaries. The boundaries are dilated by a fixed radius (6-8 pixels for a 600x800 image), then connected regions are extracted. We select the region with the maximum area (i.e. number of boundary pixels), but unlike [21], we only count edge pixels in the middle 50% of the image. This is motivated by the observation that most of the foreground is typically near the center of the image, although it does not restrict objects from being close to the image boundaries. The foreground mask corresponds to the region with the highest pixel count. We determined empirically that it is useful to fill the holes in the foreground region when dealing with sketches, but this is not necessary for textbook graphics. This approach yields 100% accuracy for the images used throughout this paper, meaning the filtering process never removes a true joint from consideration. We hypothesize that this simple step has a large impact on the quality of joint detection, which we investigate further in section 6.

5.2. *Detecting rigid bodies*

As alluded to in section 4, rigid bodies are not explicitly recognized using a sliding window detector; rather, they are identified by local features from pairs of detected joints. This decision was motivated by the observation that rigid bodies exhibit high variance in shape, color, texture, and size, making it difficult to learn a reliable detector using typical image features. Since any rigid body must be connected to at least two

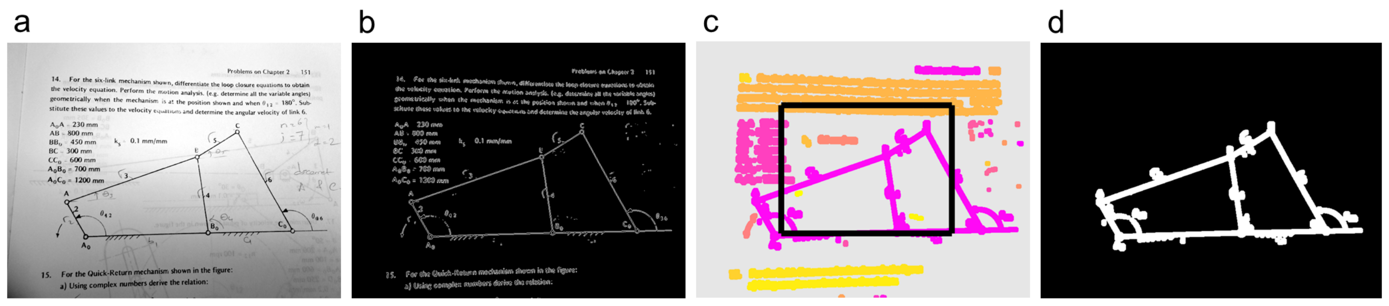

Figure 5: Greedy foreground extraction on a sample textbook image. (a) The input grayscale image after saturating the top and bottom 1% of pixels. (b) A binary mask of boundaries detected by the Sobel edge filter. (c) After dilating the boundaries by a small amount, regions are grouped together (distinguished here by different colors) and edge pixels are counted for each region. Note that only pixels inside the black bounding box are counted and the background (grey) is ignored. (d) The region with the highest count is identified as the foreground.

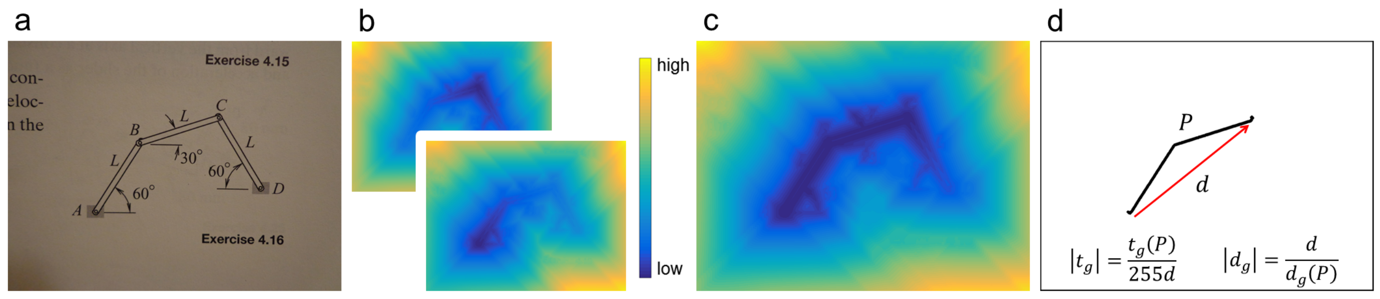

Figure 6: Computation of normalized geodesic time and distance for a pair of points in a sample image. (a) The grayscale image, from which we are comparing the pin joints at A and C. (b) Heatmaps indicate the geodesic time between pin A (*front*) or pin C (*back*) and every other pixel in the image. (c) The sum of the heatmaps can be used to extract the optimal path between the pair of points. (d) The optimal path P has an associated distance (pixel count) and time (sum of the pixel intensities along the path). The distance d between the pair of points is used to normalize the geodesic measures between 0 and 1.

joints[3], we have already reduced the search space by identifying probable joints and can now check pairs of joints for body characteristics.

In contrast to our previous work in this area [21, 22], we currently present a *supervised* learning approach for the task of body detection (g_2 in Figure 2). Specifically, we train a soft, linear SVM in the four-dimensional space defined by the following features: (i) normalized geodesic time; (ii) normalized geodesic distance; (iii) minimum joint confidence for the pair in question; and (iv) maximum joint confidence between the same joints.

The geodesic time [59] between any pair of pixels (p, q) in a grayscale image is defined as

$$t_g(p, q) = \min\{\, t_g(P) \mid P \text{ links } p \text{ to } q \,\} \tag{1}$$

and the geodesic time of a given path P of length n connecting two pixels is given by

$$t_g(P) = \frac{I_{p_0}}{2} + \frac{I_{p_n}}{2} + \sum_{i=1}^{n-1} I_{p_i} \tag{2}$$

where I_{p_j} is the pixel intensity of the jth pixel on the path. In other words, the geodesic time indicates how long it would take to traverse an image "landscape", if you followed the quickest path between points and height of the landscape is determined by pixel intensity. We believe this is a useful metric for characterizing rigid bodies in mechanical linkages because there is typically a path through dark boundary lines for joints located on the same body. As a result, we can expect the geodesic time for true pairwise connections to be low and for false connections to be high. To scale all data in the range of [0,1], we divide the geodesic time between two points by the worst-case scenario, which is a line of white pixels connecting the points. Without this normalization, the algorithm may exhibit positive bias toward false connections between joints that are near each other and negative bias toward true connections in which the joints are far apart.

On its own, normalized geodesic time is likely insufficient for discriminating between true joint connections and false connections. The reason for this is simple: mechanical linkages can be viewed as chains of connected bodies. As such, there should be a path with low geodesic time between joints at opposite ends of the chain that are not on the same rigid body. To combat this challenge, we include a second metric called normalized geodesic distance.

The geodesic distance [59] between any pair of pixels (p, q) in a grayscale image is defined as

$$d_g(p, q) = \{\, L(P) \mid t_g(P) = t_g(p, q) \,\} \tag{3}$$

where $L(P)$ is the length (number of pixels) of the minimum

[3]The exception to this rule is a pendulum, which contains a rigid body connected by only one pin; this case is excluded from the current domain of interest due to lack of complexity.

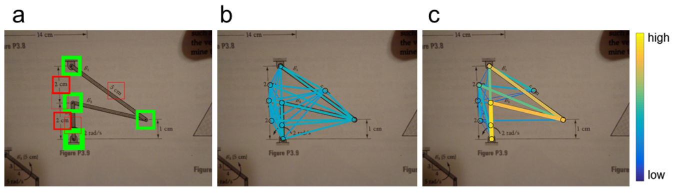

Figure 7: Comparison of rigid body detection approaches. (a) Sample image with true positive (green) and false positive (red) joint detections highlighted; (b) Previous unsupervised method based only on normalized geodesic time; (c) Current supervised approach using four features.

path P connecting p and q. To scale all data in the range of [0,1], we apply the following normalization

$$|\, d_g(p,q)\,| = \frac{||p,q||}{d_g(p,q)} \tag{4}$$

where $||p,q||$ is the Euclidean distance between p and q. Based on this formula, pairs of joints that have an approximately straight-line minimum path will yield high values for normalized geodesic distance, while pairs that require a longer, obscure path will result in low geodesic distance values. An illustrative summary of the geodesic computations is shown in Figure 6.

While low geodesic time and high geodesic distance are good indicators that a pair of joints belong to the same rigid body, there are some limitations. Consider, for instance, the example depicted in Figure 7. In this case, multiple false joint detections are located near the edge of a rigid body (Figure 7a). Thus, the normalized geodesic time and distance between these false joints and true joints will likely create false rigid body connections that are indistinguishable from true connections (Figure 7b). In order to mitigate this effect, we add pairwise joint confidences as features for the rigid body classifier. In this way, false connections may be correctly identified as such due to the low relative confidence of the false joint (Figure 7c).

5.2.1. *Training*

Using the results of the pin joint detector as context cues for rigid body detection has an important implication for training protocol. Given the sequential nature of the vision pipeline, the rigid body detector is prone to overfit the training data if all samples are used at once to train the previous detector. Ideally, we would like to train the rigid body detector using context cues that mimic joint detection behavior during testing. To accomplish this, we implement a stacked training procedure similar to [54] and described in [10, 63].

The underlying idea of stacked learning is that training data for a classifier that relies on information from a previous classifier should only be generated from images that were not used for training the previous classifier. In this sense, stacked learning is similar to cross-validation. Figure 8 demonstrates this concept. To produce training data for rigid body detection, we split the training database into k equal subsets ($k = 5$ for all experiments in this paper). Then, we train a joint detector on $(k-1)$ subsets and test it on the remaining subset. By repeating this process k times, we effectively create an unbiased set of

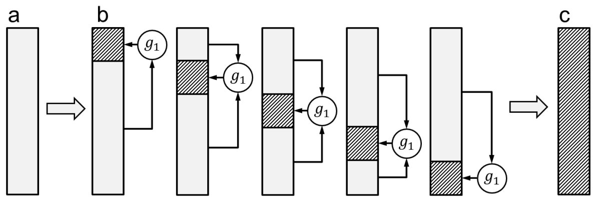

Figure 8: Stacked learning protocol. (a) The original training database is split into k folds. (b) For each fold, the g_1 classifier is trained using the remaining $(k-1)$ folds. (c) The test results for each fold are compiled into a new training database for subsequent classifiers that has the same size as the original database.

joint detections for each image in the original training database. This data is subsequently used to train the rigid body detector using the aforementioned features.

5.2.2. *Filtering unused joints*

During testing, all pairs of detected joints in an image are evaluated using the learned SVM. Similar to the joint detection protocol, the confidence of a rigid body connection is determined by its distance to the SVM decision boundary, and connections with distance less than zero are discarded. As a result, it may occur that one or more joints have no positive connections to other joints. As a rule, we remove such joints because they are no longer useful for the optimization routine (h_2 in Figure 2).

5.3. *Optimizing mechanical structure*

Given the output from the vision pipeline, which is a set of confidence values associated with detected joints and pairwise joint connections, the problem becomes one of constrained multiobjective optimization (h_3 in Figure 2). Specifically, we seek to find a hypothesis of a mechanical linkage that is strongly consistent with what has been detected in the image as well as reasonable in terms of kinematic simulation. To that end, we employ the nondominated sorting genetic algorithm (NSGA-II) introduced by Deb et al. [19]. This algorithm was selected for several reasons, including its well-known success in solving real-world applications, its ability to quickly find a diverse set of good solutions for multiple conflicting objectives, and the unique opportunities it provides for handling constraints. Notable details specific to the present domain are outlined below.

5.3.1. *Representation*

We represent a mechanical linkage using a structured graph, similar to [44], but with vertices representing joints and edges corresponding to pairs of joints on the same rigid body. It is important to note that edges do not explicitly identify rigid bodies (i.e. there is not a one-to-one correspondence); in fact, any body containing more than two joints will be encoded as a fully connected subgraph comprising multiple vertices and edges. In addition, we relax the constraint that multiple pins fixed to the ground must share an edge because a typical image may not exhibit strong visual cues that grounded pins are connected. Consequently, the graph representation of a given mechanism

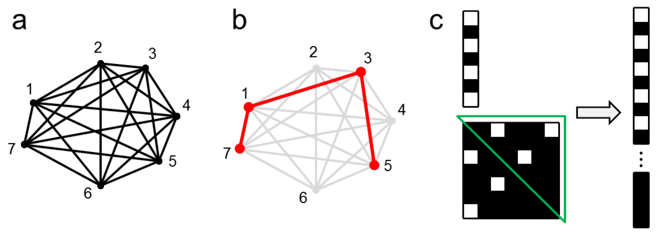

Figure 9: Phenotype-genotype mapping. (a) An artificial example of the phenotype space comprising all available graph vertices and edges. Note that the graph does not have to be fully connected as shown here. (b) A candidate solution, highlighted in red, contains a subset of the available graph. (c) The first part of the genotype encodes which vertices exist in the solution, and the second part dictates which edges should be included. The latter portion is derived from the upper triangular adjacency matrix.

is not necessarily unique; also, multiple linkages with different kinematic behavior can be represented by the same graph.

Figure 9 illustrates the phenotype-genotype mapping for our problem. The genotype is a vectorized transformation of the structured graph, and it consists of two parts: (i) a bit-string of length N to indicate the presence of a joint in an individual hypothesis, where N is the number of detected joints; and (ii) a bit string of length $N(N-1)/2$, which encodes the presence of pairwise edges. The latter part is derived from the upper triangular matrix of the adjacency matrix, which is symmetric. It should be noted that the genotype does not explicitly include any information regarding the spatial layout of detected joints. Also, this representation allows an edge to exist in the absence of one or more of its connecting vertices. To deal with this issue, the bit-string that encodes vertices (joints) is used to virtually mask "invalid" edges (connections) for the purpose of fitness evaluation. However, the true value of those edges is retained during crossover and mutation so that there is no internal bias toward solutions with fewer edges. Similarly, this representation allows a vertex to exist in the absence of one or more of its connecting edges. We virtually mask such vertices during fitness evaluation, but keep the true value during crossover and mutation.

5.3.2. Fitness criteria

The ultimate goal of our recognition framework is to find the mechanism topology graph that concurrently optimizes a series of objectives using Pareto dominance. Specifically, we seek to maximize the image-based confidence of joints and connections used in a solution, to minimize the confidence of unused image information, and to maximize the likelihood that the mechanical behavior is meaningful.

1. The *joint likelihood* (f_1) is the average detection confidence of joints present in a given hypothesis (H), or

$$f_1(H) = \frac{\sum_{i=1}^{N} s_{j,i} y_i}{\sum_{i=1}^{N} y_i} \tag{5}$$

   where $s_{j,i}$ is the confidence of the ith joint and y_i is the (Boolean) value of the ith gene.

2. Similarly, the *joint connection likelihood* (f_2) for a given hypothesis is computed as the average connection confidence,

$$f_2(H) = \frac{\sum_{i=N+1}^{L} s_{c,i} y_i}{\sum_{i=N+1}^{L} y_i} \tag{6}$$

   where $s_{c,i}$ and L are the confidence of the ith connection and the chromosome length, respectively.

3. *Residual image data* refers to the average confidence of unused joints (f_3) and connections (f_4) in an individual solution. These two objectives are critical for enabling the detection of complex mechanisms; without them, a four-bar linkage (the simplest mechanism in our domain) will *always* be preferred unless additional nodes and edges improve the confidence of f_1 or f_2.

4. A series of binary *mechanical constraints* are evaluated to estimate the kinematic feasibility of solutions. The percentage of constraints that are satisfied make up the final objective (f_5); for a feasible mechanical linkage, this will equal one. No constraint is more important than the others, and the feasibility objective contributes to nondominated sorting in the same way as the other fitness criteria.

5.3.3. Mechanical constraints

In this section, we discuss the design of the mechanical constraints on feasibility in more detail since this is fundamentally the most critical objective function; without it, the optimization simply maximizes object detection output with no regard to the domain of interest. By *feasible*, we mean that a solution should represent a single, closed kinematic chain and that all rigid bodies (with the exception of a world frame) should be capable of motion. While it may be easy for an expert engineer to evaluate these criteria, it is a nontrivial task to transform them into mathematical constraints that can be used by the computer. Ideally, a full kinematic simulation would provide the best insight regarding feasibility, but this approach is too computationally expensive (imagine generating 10,000+ dynamic simulations quickly). Instead, we found through trial-and-error that simple heuristics yield a reasonable estimate of feasibility without the lag in computational speed.

The initial set of constraints we developed [22] include: (i) the degrees of freedom (DOF) should be greater than zero, (ii) each joint must have at least one connection, (iii) there must be at least four rigid bodies, including the frame, and (*iv*) all joints must be a minimum distance away from each other (e.g. 15-30 pixels). Even though each of these constraints is indeed a requirement for a mechanical linkage to be feasible, our implementation was perhaps too simplistic. For example, we used Gruebler's equation [60] to compute the DOF, such that

$$DOF = 3(n_b - 1) - 2n_{lp} - n_{hp} \tag{7}$$

where n_b is the number of rigid bodies (including the world frame), n_{lp} is the number of lower kinematic pairs (i.e. revolute and prismatic joints), and n_{hp} is the number of higher

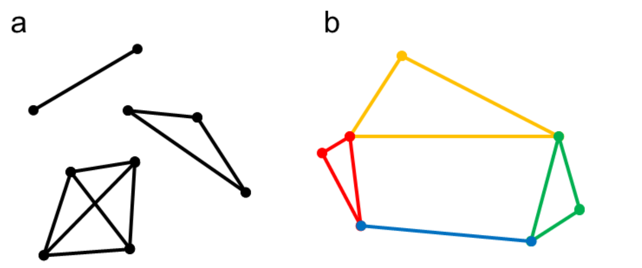

Figure 10: (a) Example cliques containing two, three, and four vertices, respectively. (b) A graph with thirteen cliques, but only four maximal cliques (distinguished by color).

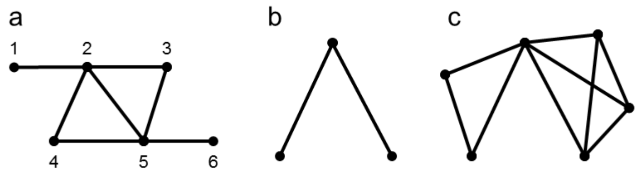

Figure 11: Mechanical constraint violations. (a) In this graph, a clique $\{2,5\}$ participates in more than one maximal clique ($\{2,3,5\}, \{2,4,5\}$). (b-c) In these instances, two maximal cliques form a subgraph with zero degrees-of-freedom (i.e. if you fix the free ends, the cliques cannot move relative to each other). Note that both graphs violate the same constraint despite having different topologies.

kinematic pairs[4]. However, we substituted the number of edges for n_b, which, as explained previously, overestimates the true rigid body count. This is also problematic for the third constraint listed above. The effect of this superficial computation on the optimization results was clear during testing; many of the Pareto-optimal solutions were not actually sensible mechanical linkages, even though they met all of the constraints we imposed.

To address this shortcoming, a significant contribution of this paper is the introduction of a new set of robust constraints that more accurately reflect mechanical feasibility. The constraints are derived from graph theory, largely from the idea of cliques [31]. A *clique* is a fully connected subgraph – that is, all pairs of vertices in the clique must have an edge between them. See Figure 10 for an illustration of this concept. A *maximal clique* is one that is not contained within a larger clique. So, for example, the graph in Figure 10b has thirteen cliques, but only four maximal cliques. Mechanically speaking, a fully connected set of links connected by pins cannot move relative to each other. Therein, locating the rigid bodies in a graph amounts to finding all of the maximal cliques [51]. There are three important constraints related to cliques (rigid bodies). First, there must be at least three maximal cliques for a feasible mechanical linkage with only revolute joints; four rigid bodies are required[5], including the world frame, which may not be explicitly identified. Second, no cliques can be members of more than one maximal clique (Figure 11a), for this violates the definition that all pairs of joints sharing a rigid body should be connected. Third, no rigid bodies (maximal cliques) can form a subgraph with zero DOF; the prime example of this violation is the triangular graph shown in Figure 11b-c, which if you assume the world frame connects the bottom two vertices, cannot move at all. See Appendix A for details on the computation of these three constraints. As a final constraint, because of the nature of kinematic chains, the graph should be connected – that is, there must exist a path between every pair of vertices. To determine whether this is true for a given solution, we use the Dulmage-Mendelsohn decomposition of the adjacency matrix [53].

To recap, the updated mechanical constraints for feasibility are as follows:

1. There must be at least three maximal cliques.
2. There must be no cliques that are members of more than one maximal clique.
3. There must be no subgraphs with zero DOF.
4. There must be only one connected component.

6. Experiments

6.1. Data

For textbook graphics, we use the MECH135 dataset [22], which consists of 135 images of planar mechanical linkages containing only revolute joints. For hand-drawn sketches, we use the MECHS250 dataset [21], which comprises 250 images of sketches created by 25 engineering students (10 sketches per user). Each sketch depicts a single planar mechanical linkage chosen randomly from the MECH135 dataset. All images are scaled to 600x800 pixels.

6.2. Methods

The primary goal of the experimental studies presented here is to characterize the performance of each subroutine in our recognition framework. The three subroutines of interest are joint detection (section 5.1), body detection (section 5.2), and structural optimization (section 5.3). Previous work [21, 22] only focused on evaluation at the optimization end, which limited our ability to quantify the impact of earlier stages in the pipeline when comparing algorithm modifications. In this paper, we systematically evaluate each stage for all combinations of detections schemes and optimization constraints to determine which methods yield state-of-the-art performance and to diagnose areas requiring improvement in the future. Table 1 summarizes the algorithm variants explored in this paper.

We conduct two sets of independent experimental studies, one for textbook graphics and one for hand-drawn sketches. The former includes three textbooks for testing, and the latter tests sketches from three users. Considerable effort was made to select exemplary users with varied sketching style and level of abstraction. Unlike our previous intermodal study [21], in this

[4]A higher kinematic pair is one in which the contact between bodies is along a point or a line; lower kinematic pairs have contact along a plane. None of the mechanisms studied in this paper have higher kinematic pairs, so this last term can be ignored.

[5]To verify this, note that a closed kinematic chain with m bodies requires at least m pin joints, and then try plugging values of m into n_b and n_{lp} in Equation (7). Only values >=4 will yield $DOF > 0$.

Table 1: Description of algorithm variants

Joint Detection	
SVM	fixed-window, soft, linear support vector machine trained on HOG features
CSVM	cascade of three context-based support vector machines trained on HOG features
SVM+FG	same as SVM, but with the additional foreground filter
CSVM+FG	same as CSVM, but with the additional foreground filter
Body Detection	
SVM4	support vector machine trained on four features including normalized geodesic distance and time as well as confidence of detected joints
SVM2	same as SVM4, except that joint detection confidences are ignored
$\lvert d_g \rvert$	normalized geodesic distance (unsupervised)
$\lvert t_g \rvert$	normalized geodesic time (unsupervised)
Mechanical Constraints for Structural Optimization	
NAIVE	initial set based on DOF, number of joint connections, and distances between joints
GRAPH-THEORETIC	updated set based on graph theory

Table 2: General NSGA-II parameters

Parameter	Symbol	Value
population size	μ	$200N^{\dagger}$
number of offspring	λ	μ
maximum number of generations	n	20
crossover method	–	uniform
crossover probability	p_c	0.9
mutation method	–	uniform
mutation probability	p_m	0.1
tournament size	k	0.02μ

$^{\dagger}N$ refers to number of detected joints

paper we exclusively train and test our framework on the same type of image at a given time – that is, textbook graphics are used to train a system that is tested on textbook graphics and sketches are used to train a system that is tested on sketches. Nonetheless, to avoid overfitting, we do not allow testing images to be involved during the training process. In fact, we do not even allow images from the same subset (e.g. textbook, user) to be included in both training and testing. In this way, we believe our protocol mimics the practical scenario of a student or engineer using a pre-trained recognition framework on a new textbook or previously unseen sketches.

For a given experimental condition and test subset, we train a joint detector using the remaining images in the corresponding dataset (MECH135 or MECHS250). As needed, we train a body detector using 5-fold stacked learning (section 5.2.1). The detectors are used to locate probable joints and pairwise joint connections in every test image. Then, for each image, NSGA-II is run five times using the general parameters listed in Table 2 and the appropriate mechanical constraints. We follow the measures against uncertainty and post-processing methods described in [21, 22]. The full implementation has been developed in MATLAB [3], and all experiments were performed on an Intel(R) quad-core 3.30GHz CPU with 8 GB RAM.

6.3. Evaluation metrics

In this section, we highlight relevant performance measures for each stage in our recognition framework. For joint and body detection, we employ Precision-Recall (PR) curves [56] because they are a well-established assessment technique used in computer vision [16]. For instance, PR curves are the principal method for comparing algorithms in the popular PASCAL VOC challenge [23]. As the name implies, a PR curve is a plot of precision versus recall for a binary classifier. *Precision* is defined as

$$Pr = \frac{tp}{tp + fp} \tag{8}$$

and *recall* is given by

$$Re = \frac{tp}{tp + fn} \tag{9}$$

where tp is the number of true positives (correctly identified objects), fp is the number of false positives (false alarms), and fn is the number of false negatives (missed objects). Precision reflects the number of false detections made by the classifier, or the positive predictive value (i.e. How confident can we be that a detection is actually a good one?). Recall reflects the sensitivity of the classifier to missed detections (i.e. How confident can we be that the classifier will not miss desired detections?). Both precision and recall can vary from 0 to 1, with 1 being the best score. In other words, the globally optimal operating point in Precision-Recall space is at (1,1), or the top-right corner of the plot.

A PR curve is generated as follows. For a set of detections, perform ground truth matching to locate true matches. Then, sort the detections based on confidence. Vary a threshold between the minimum and maximum confidence values. At each threshold, consider all detections with confidence greater than the threshold to be positive detections and all others negative. The relevant variables (tp, fp, fn) can be computed from the positive/negative labels and ground truth matches, and a single value for precision and recall is calculated. After computing the precision and recall at each threshold, the result can be plotted.

A PR curve can provide qualitative assessment of the strengths and weaknesses of a classifier. For quantitative comparison, most researchers use the average precision (AP), or the area under the curve. In our experiments, the mean average precision (mAP) across different textbooks or users yields a holistic view of classifier performance.

For mechanical graph optimization, we leverage a collection of different evaluation metrics to assess overall performance. First, we look at whether an image is *solvable*, meaning the true graph is contained within the search space of the optimization. In a way, the percentage of images that are solvable reflects the combined quality of the joint and body detection schemes. If the joint detector fails to locate one of the true joints in an image, for instance, the optimization will *never* evolve the true solution. The next metric we consider is the percentage of *solved* images. By this we mean the number of images for which the true solution was found in at least one independent run. The relative difference between the percentage of solvable and solved images may provide insight into the limits of the optimization. An image that is solvable, but not solved, may indicate that the joint and body detectors have high recall, but low mean average precision. In other words, weak confidence in true objects or

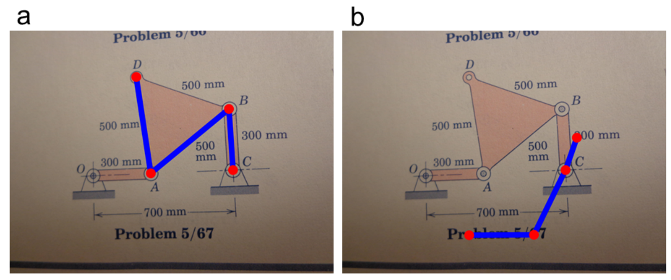

Figure 12: Both of these solutions do not match the ground truth, but are they equally bad? In (a), the solution can be corrected in one step by moving the joint at point D to point O. The solution pictured in (b) requires multiple steps to accurately match the ground truth.

strong confidence in false objects may restrict the ability of the optimization routine to evolve the true solution. Refer to [21] for additional details and examples of unsolvable and unsolved images.

Another performance measure used to assess optimization quality is top-N accuracy. Top-N accuracy refers to the percentage of runs in which the true solution was located in at least the top N sorted Pareto-optimal solutions. Ideally, we desire for the true solution to be the top solution (high top-1 accuracy), but this may not be realistic given the fact that we have multiple, conflicting objective functions. As a result, we also look at top-5 accuracy and consider post hoc user selection or full kinematic simulation of the top 5 solutions to be reasonable approaches for narrowing the output set of solutions.

In our baseline experiments [21, 22], we relied on top-N accuracy to measure the success of our framework. While it is a valuable metric, it may be too coarse in resolution. Consider the examples in Figure 12. Neither of these graphs matches the ground truth, but it is clear that the graph on the left is closer to the true solution than the one on the right. Is there a way to capture this information? We propose a novel evaluation metric called the *user effort ratio*. The basic premise is that the worst recognizer would require the user to manually create the entire model on their own (similar to what current modeling software requires). The best recognizer, on the other hand, would not require any effort from the user for model creation. In practice, our recognition framework likely exists somewhere in the middle. It may not always get the true mechanical linkage correct, but it should at least give results that are close to the true solution. In this way, it is conceivable that the user could correct the model with simple, intuitive strokes (e.g. add a missing joint by circling it, remove a false edge by crossing it out).

The user effort ratio (R_E) is thus defined as

$$R_E = \frac{\text{number of steps to correct model}}{\text{number of steps for manual model creation}} \tag{10}$$

Lower values of R_E are preferred. The denominator is easily computed as the sum of all vertices and edges in the ground truth solution; the user would simply need to specify each one to construct the graph manually. The numerator, however, is treated a bit differently. As illustrated in Figure 13, we consider each of the following user operations to count as one step: (i) adding a joint; (ii) adding an edge between two existing joints; (iii) removing a joint and its corresponding edges; (iv) removing an edge; and (v) moving the location of a joint, keeping its edges intact. This novel performance measure has two advantages. First, it provides more fine-grained information for comparing optimization routines. Second, it gives an interesting estimate of how beneficial our computational method will be to end users – that is, how much effort will our automatic recognition framework save them? In this paper, we compute user effort ratio based solely on the top solution, although in theory an average user effort ratio could be computed on a small set (e.g. top 5 solutions).

Table 3: Average Precision for joint detection in textbook graphics

	SVM	CSVM	SVM+FG	CSVM+FG
Ginsberg	0.48	0.79	0.75	0.88
McGillKing	0.89	0.88	0.96	0.95
MeriamKraige	0.91	0.91	0.92	0.93
mAP	0.76	0.86	0.88	0.92

6.4. *Results and discussion*

Sample results for both textbook graphics and hand-drawn sketches are provided in Figure 14. In the following sections, we discuss the results for each study separately.

6.4.1. *Textbook graphics*

The three textbooks selected for testing are referenced by author name: Ginsberg [30], McGillKing [47], and MeriamKraige [48]. Each textbook contributes 3, 15, and 27 images to the dataset, respectively. Precision-recall curves for joint detection are shown in Figure 15 and average precision results are summarized in Table 3. In general, mean average precision is high (>0.75 for all cases with a maximum of 0.92 for the CSVM+FG case), indicating that the joint detection methods we selected are largely successful. It should be noted that the results for Ginsberg appear noisier than the others, but this is simply due to the low number of images for that textbook (3) and is not representative of large-scale detection performance. Still, all textbooks demonstrate the same trend of increasing mAP as multiple context classifiers are added and even more so when the foreground filter is applied. The markers on each plot in Figure 15 pinpoint the actual operating points of each classifier, and they exemplify the same trend. An important outcome of this assessment is that recall is always very high, meaning we rarely miss a joint detection, even if there are many false positives. Since it is clear that including foreground extraction always yields better average precision, we only consider the SVM+FG and CSVM+FG cases further in the pipeline.

Precision-recall curves for rigid body detection are illustrated in Figure 16 and accompanied by average precision results in Table 4. For this, we analyze eight experimental conditions based on the two remaining joint detection schemes and the four available body detection methods. The mean average precision is markedly lower than the joint detection mAP (Table 3) for all

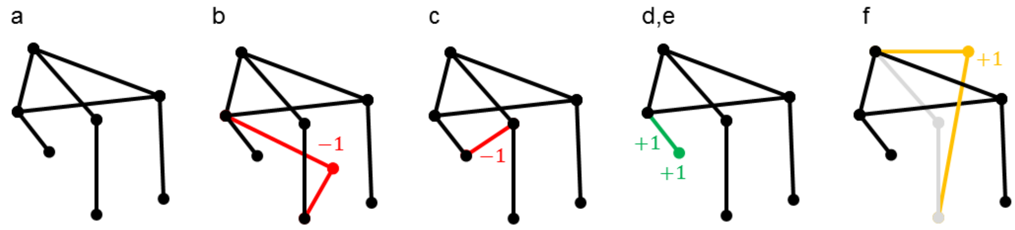

Figure 13: Artificial examples of user effort required to fix an incorrect solution. (a) The correct solution. Each of the following interactions count as one step: (b) removing a joint and its associated connections; (c) removing an edge between two joints; (d) adding a joint; (e) adding an edge between two joints; and (f) moving a joint to a new location, keeping its edges intact.

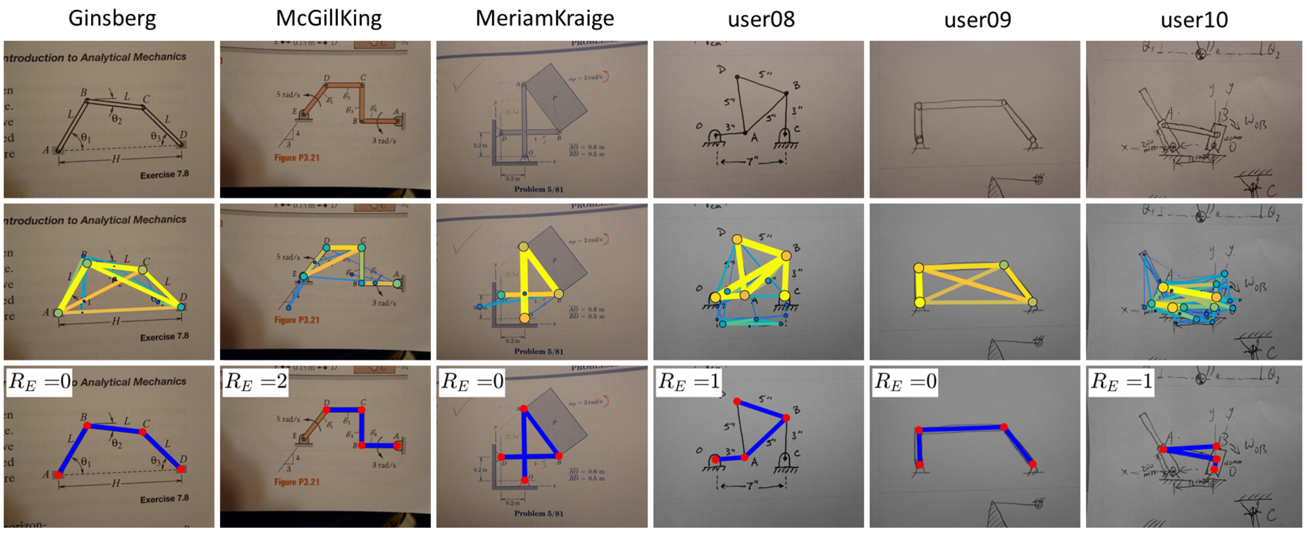

Figure 14: Sample results on textbook images and sketches. For each textbook and user, we select a representative example and show the original image (top), the output from our vision pipeline (middle), which depicts detected joints and bodies (thicker, more yellow lines and circles indicate higher confidence), and the top solution evolved during one optimization run (bottom). All solutions shown here require 2 or fewer steps for correction.

experimental conditions. This is most likely a result of the fact that our rigid body detection space depends on the joint detector. Some false positive joints may create pairwise connections with very high confidence. Also, whereas our joint detector is trained on a high-dimensional (1764) feature space, the body detector uses at most four features, which may not be discriminative enough. Nonetheless, the results exhibit some important trends. While the difference that is attributable to joint detection is small, the SVM4 method outperforms all other body detection methods. Interestingly, the unsupervised approaches do not fare well, and the method that is most similar to our prior work (geodesic time [22]) has the lowest mAP by far. Knowing this, we only include the SVM4 and SVM2 conditions in further analyses.

Figure 17 depicts the performance of our recognition framework at the optimization stage. The eight remaining experimental conditions are described in Table 5. Solvability is high ($> 90\%$) for all cases, which reemphasizes that the combined vision-based detectors generate few false negatives. The percentage of solved images, however, varies significantly. For this metric, we can draw the following conclusions: (i) the new graph-theoretic mechanical constraints perform better than the old naive constraints (compare $1 \leftrightarrow 5$, $2 \leftrightarrow 6$, $3 \leftrightarrow 7$, $4 \leftrightarrow 8$); (ii) SVM4 leads to more solved images than SVM2 (compare 1↔3, 2↔4, 5↔7, 6↔8); and (iii) the joint detection method that works better depends on the body detection method (compare 1↔2, 3↔4, 5↔6, 7↔8). Sometimes, CSVM+FG outperforms SVM+FG, but the condition with the highest overall percentage of solved images uses SVM+FG (condition 5).

There are two meaningful measures of top-N accuracy, one related to performance on *any* image and one related to performance only on *solved* images. These metrics are indicated in Figure 17 by darker and lighter bars, respectively. One way to think about this distinction is that top-N accuracy on all images provides a sense for the overall success of the optimization, while top-N accuracy on solved images represents the repeatability of the optimization. For top-1 accuracy, the naive constraints do better on solved images (99.3% for condition 1), but the graph-theoretic constraints do better overall (70.2% for condition 5). Top-5 accuracy shows a similar trend, except that the percentages for solved images are now relatively even among all conditions.

Perhaps the most important takeaway of this work is the *user effort ratio*. The results in Figure 17 are very promising. Keeping in mind that lower values indicate less required effort from the user, the best experimental condition is

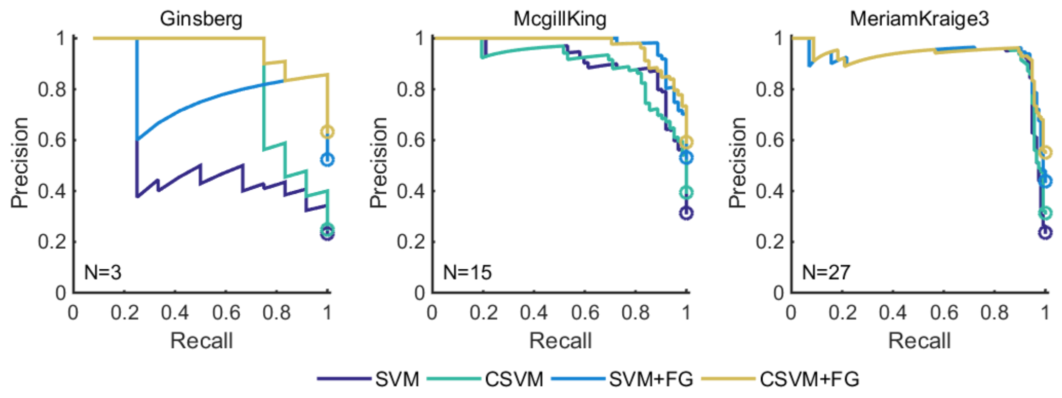

Figure 15: Precision-Recall curves for joint detection in textbook graphics.

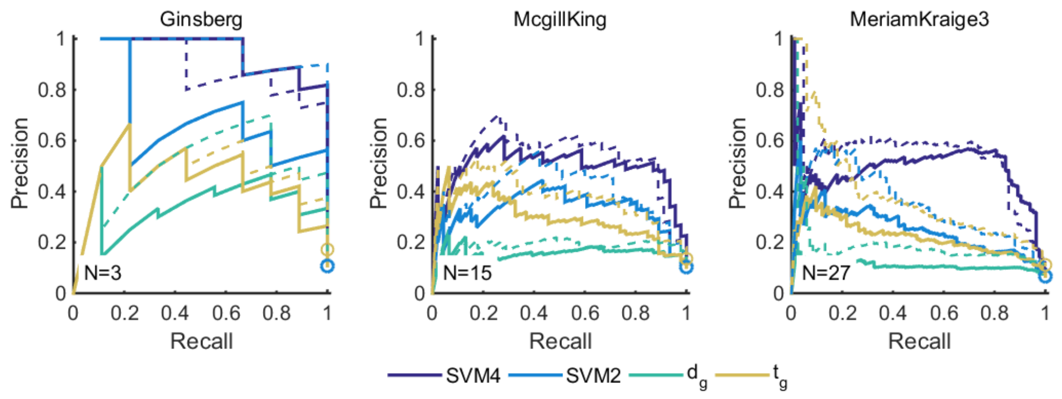

Figure 16: Precision-Recall curves for body detection in textbook graphics. Solid lines represent the SVM+FG joint detection method; dashed lines refer to CSVM+FG.

Table 4: Average Precision for body detection in textbook graphics

Body Detection	**SVM4**		**SVM2**		**$\|d_g\|$**		**$\|t_g\|$**	
Joint Detection	**SVM+FG**	**CSVM+FG**	**SVM+FG**	**CSVM+FG**	**SVM+FG**	**CSVM+FG**	**SVM+FG**	**CSVM+FG**
Ginsberg	0.84	0.78	0.59	0.85	0.34	0.49	0.06	0.10
McGillKing	0.49	0.52	0.32	0.39	0.16	0.19	0.06	0.08
MeriamKraige	0.48	0.53	0.25	0.34	0.11	0.18	0.04	0.06
mAP	0.60	0.61	0.38	0.53	0.20	0.29	0.05	0.08

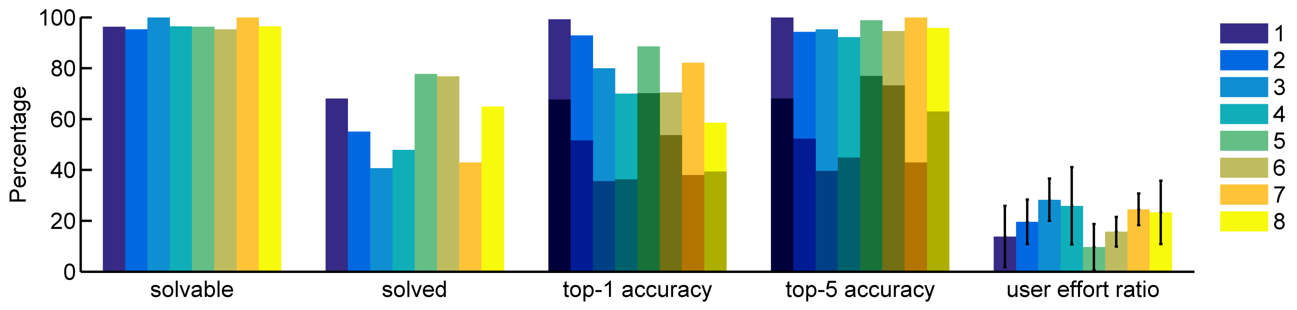

Figure 17: Optimization results on textbook graphics using eight experimental conditions. **Solvable**: average percentage of images from a textbook for which the optimization *can* find the correct solution; **Solved**: average percentage of images from a textbook for which the optimization *does* find the correct solution in at least one run; **Top-*N* accuracy**: average percentage of images from a textbook for which the optimization finds the correct solution in the top *N* solutions; darker bars account for all images, while lighter bars only include accuracy for *solved* images; **User effort ratio**: average amount of effort required by a user to correct the top solution for images from a given textbook; error bars on reveal one standard deviation.

Table 5: Experimental conditions for optimization

ID	Joints	Bodies	Constraints
1	SVM+FG	SVM4	NAIVE
2	CSVM+FG	SVM4	NAIVE
3	SVM+FG	SVM2	NAIVE
4	CSVM+FG	SVM2	NAIVE
5	SVM+FG	SVM4	GRAPH-THEORETIC
6	CSVM+FG	SVM4	GRAPH-THEORETIC
7	SVM+FG	SVM2	GRAPH-THEORETIC
8	CSVM+FG	SVM2	GRAPH-THEORETIC

Table 6: Average Precision for joint detection in hand-drawn sketches

	SVM	CSVM	SVM+FG	CSVM+FG
user08	0.61	0.59	0.81	0.82
user09	0.93	0.94	0.97	0.97
user10	0.65	0.73	0.75	0.83
mAP	0.73	0.75	0.84	0.87

5 (SVM+FG/SVM4/GRAPH-THEORETIC) with a *user effort ratio* of 9.7%. For all experimental conditions, our recognition framework is effectively reducing user effort by at least 75%.

6.4.2. Hand-drawn sketches

Next, we applied the same experimental conditions to images of hand-drawn sketches from three users. Figure 18 shows precision-recall curves for joint detection, and Table 6 lists average precision results. For any given method, the mAP is reasonably high, but 6.7% lower on average than the corresponding mAP for textbooks. Precision at the operating point has high variance and is most likely dependent on sketching style. The PR curves for joint detection exemplify the same trends for sketches as for textbooks (i.e. CSVM+FG > SVM+FG > CSVM > SVM).

Precision-recall curves for rigid body detection in sketches are illustrated in Figure 19 with average precision results in Table 7. Similar to textbook graphics, mean average precision is lower for bodies than for joints and the SVM4 method greatly outperforms the other approaches. The CSVM+FG joint detection leads to slightly higher mAP for bodies than SVM+FG, but the difference is not significant.

Again, we only consider the eight experimental conditions outlined in Table 5 for evaluating the optimization stage. The results are illustrated in Figure 20. Generally speaking, the recognition framework performs reasonably well on sketches, although not as well as textbooks. Solvability is high (96.7%) and is unaffected by choice of detection scheme or mechanical constraints. As with textbook graphics, condition 5 is able to solve the most images, but the percentage is lower (40%). Regarding top-*N* accuracy and the *user effort ratio*, the difference in using SVM+FG versus CSVM+FG for joint detection appears to be small. The new mechanical constraints seem to positively impact performance, but the largest improvement comes from using the SVM4 method for detecting bodies over the SVM2 method. One significant result from Figure 20 is that conditions 5 and 6 have top-5 accuracy of 100% for solved images. In other words, even though the number of images that are solved may be low, the optimization is *always* able to evolve the correct solution in the top five candidates for images that it does solve under those two conditions. The user effort required on average to correct the top solution is 27-28% for all conditions using SVM4 (1, 2, 5, and 6). While it may seem redundant to correct an image of a paper-and-pencil sketch with digital sketch-based gestures, we believe our framework may be particularly useful if both sketch creation and editing operations were digital.

6.4.3. Limitations

There are two major limitations of this work in its current state. First, the optimization operates under the assumption that all true joints are at least weakly detected; in the presence of a false negative, this method will never find the correct solution. Solving this issue will likely require something similar to expectation-maximization, in which detected joints and bodies iteratively inform the likelihood of each other so that previously undetected joints can be identified if there is the strong presence of a rigid body nearby. Second, the current domain is limited in scope. We did this purposefully to ensure our problem was tractable; however, future work should look to extend this framework to more complex mechanical behaviors (e.g. prismatic joints, gears, cams, intermittent contact).

7. Conclusions

The computational method presented here leverages well-known computer vision techniques for object recognition with evolutionary methods for optimizing the graphical structure of planar mechanical linkages in images. We conducted a thorough evaluation on textbook graphics and hand-drawn sketches to demonstrate the efficacy of each stage in the framework. Overall performance is improved by a new supervised learning method for detecting rigid bodies that takes into account the context of detected joints. In addition, new constraints on mechanical feasibility based on graph theory ensure that Pareto-optimal solutions are actually capable of exhibiting meaningful kinematic behavior. Finally, we introduce a novel evaluation metric called *user effort ratio* that reflects the benefit derived from using our automatic approach over manual model creation. The results demonstrate that with very little effort (10% for textbook graphics and 27% for sketches), a user can quickly generate kinematic models of planar mechanical linkages.

Appendix A. Computing graph-theoretic mechanical constraints

For efficient computation of maximal cliques, we use the Bron-Kerbosch method with pivoting [8, 11]. The output is an *m*-by-*n* matrix *C*, where *m* is the number of vertices (i.e. joints in a mechanical linkage) and *n* is the number of maximal cliques in the graph. Each column represents a maximal clique, encoded as a bit-string such that

$$C_{ij} = \begin{cases} 1 & \text{if vertex } i \text{ is in maximal clique } j \\ 0 & \text{otherwise} \end{cases} \tag{A.1}$$

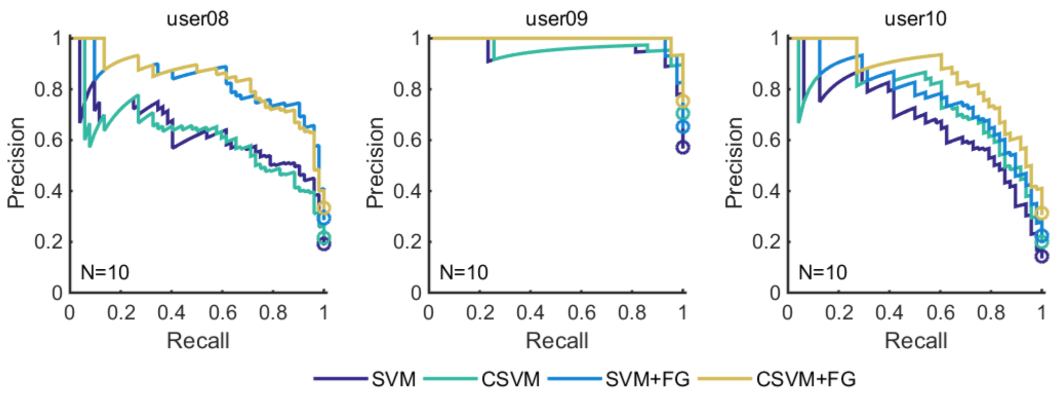

Figure 18: Precision-Recall curves for joint detection in hand-drawn sketches.

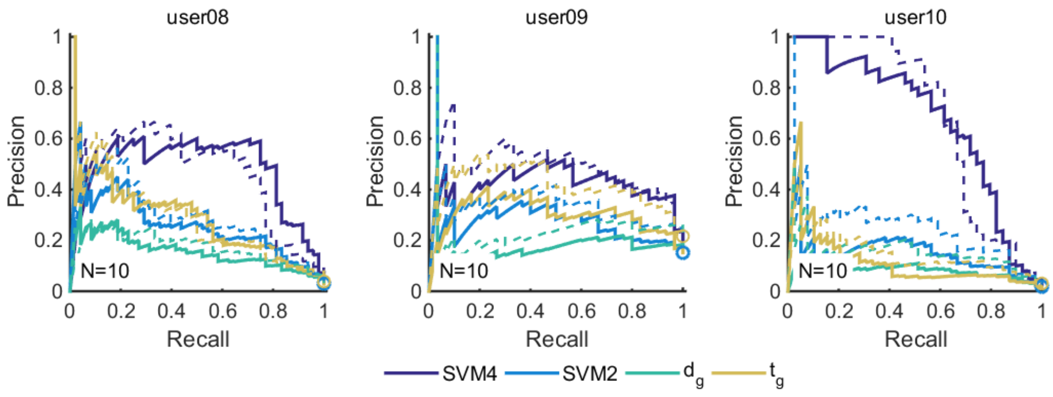

Figure 19: Precision-Recall curves for body detection in hand-drawn sketches. Solid lines represent the SVM+FG joint detection method; dashed lines refer to CSVM+FG.

Table 7: Average Precision for body detection in hand-drawn sketches

Body Detection	**SVM4**		**SVM2**		**$\|d_g\|$**		**$\|t_g\|$**	
Joint Detection	**SVM+FG**	**CSVM+FG**	**SVM+FG**	**CSVM+FG**	**SVM+FG**	**CSVM+FG**	**SVM+FG**	**CSVM+FG**
user08	0.47	0.46	0.26	0.28	0.15	0.17	0.26	0.27
user09	0.41	0.47	0.26	0.32	0.16	0.23	0.30	0.41
user10	0.65	0.66	0.15	0.23	0.07	0.15	0.11	0.15
mAP	0.51	0.53	0.22	0.28	0.13	0.18	0.22	0.28

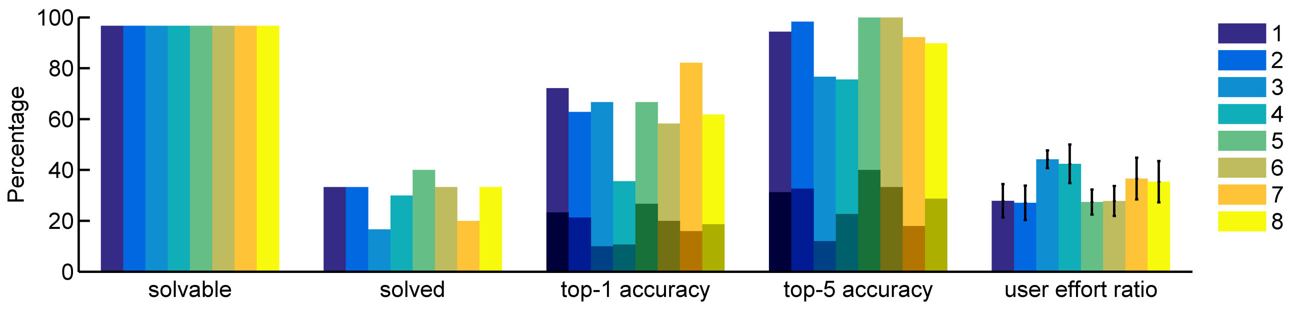

Figure 20: Optimization results on hand-drawn sketches using eight experimental conditions. **Solvable**: average percentage of images from a user for which the optimization *can* find the correct solution; **Solved**: average percentage of images from a user for which the optimization *does* find the correct solution in at least one run; **Top-*N* accuracy**: average percentage of images from a user for which the optimization finds the correct solution in the top *N* solutions; darker bars account for all images, while lighter bars only include accuracy for *solved* images; **User effort ratio**: average amount of effort required by a user to correct the top solution for images on average; error bars on reveal one standard deviation.

Next, we describe how to use the matrix C to compute each of the three relevant constraints.

1. To compute the number of maximal cliques in the graph, simply count the number of columns in C.
2. To determine whether any cliques are members of more than one maximal clique, we need to find shared edges between maximal cliques. We can do this by counting the number of shared vertices between pairs of maximal cliques, which is efficiently computed as the product of the transpose of C with itself. Let $A = C^tC$. Then, A is a symmetric, n-by-n matrix and A_{jk} equals the number of shared vertices between the jth and kth maximal clique. If any of the off-diagonal terms are greater than one, by definition this indicates one or more shared edges between a pair of maximal cliques. So, to evaluate this constraint, simply check that $A_{jk} \leq 1 \; \forall j \neq k$.
3. To check if any subgraphs have zero DOF, the primary subgraph we search for is two cliques that shared a vertex, but do not share any other vertices with each other or another clique. To determine if this subgraph exists, first we find the set of vertices that are only involved in one maximal clique,

$$\left\{ v \in \mathbb{R}^m \mid \sum_{j=1}^{m} C_{vj} = 1 \right\} \tag{A.2}$$

Next, let $B = CC^t$. B is a symmetric, m-by-m matrix in which B_{ik} equals the number of maximal cliques containing at least vertex i and vertex k. Taking the column-wise sum of the rows indexed by v yields a vector u comprising the number of vertices in v that participate in a maximal clique with each of the other vertices. If any element of u is greater than one, the triangular subgraph exists and the constraint is violated.

Let us test an example to demonstrate one of these constraint violations. For the graph in Figure 11, it is straightforward to determine that there are four maximal cliques and C can be written as

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.3}$$

The matrix of shared vertices is then given by

$$A = C^tC = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 1 & 3 & 2 & 1 \\ 1 & 2 & 3 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix} \tag{A.4}$$

Since $A(2,3) > 1$, the second and third maximal cliques (indexed by columns in C) share a clique and the second mechanical constraint is violated.

[1] *Adams, version 2013.2.* MSC Software Corporation, Newport Beach, California, 2010.

[2] *ForceEffect Motion.* Autodesk, Inc., San Rafael, California, 2014.

[3] *MATLAB, version 8.3.0.532 (R2014a).* The MathWorks Inc., Natick, Massachusetts, 2014.

[4] *Working Model 2D.* Design Simulation Technologies, Inc., Canton, Michigan, 2014.

[5] S. Acharyya and M. Mandal. Performance of eas for four-bar linkage synthesis. *Mechanism and Machine Theory*, 44(9):1784–1794, 2009.

[6] B. D. Adelstein. Three degree of freedom parallel mechanical linkage, Oct. 6 1998. US Patent 5,816,105.

[7] J. Angeles and J. Angeles. *Fundamentals of robotic mechanical systems*, volume 2. Springer, 2002.

[8] C. Bron and J. Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[9] J. Cabrera, A. Simon, and M. Prado. Optimal synthesis of mechanisms with genetic algorithms. *Mechanism and machine theory*, 37(10):1165–1177, 2002.

[10] V. Carvalho and W. Cohen. Stacked sequential learning. *Proceedings of the IJCAI-05, Edinburgh, Scotland*, 2005.

[11] F. Cazals and C. Karande. A note on the problem of reporting maximal cliques. *Theoretical Computer Science*, 407(1):564–568, 2008.

[12] C. A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer methods in applied mechanics and engineering*, 191(11):1245–1287, 2002.

[13] G. Costagliola, M. De Rosa, and V. Fuccella. Recognition and autocompletion of partially drawn symbols by using polar histograms as spatial relation descriptors. *Computers & Graphics*, 39:101–116, 2014.

[14] D. Crandall, P. Felzenszwalb, and D. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *Computer Vision and Pattern Recognition (CVPR), 2005 IEEE Conference on*, volume 1, pages 10–17. IEEE, 2005.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR), 2005 IEEE Conference on*, volume 1, pages 886–893. IEEE, 2005.

[16] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

[17] R. Davis. Magic paper: sketch-understanding research. *Computer*, 40(9):34–41, 2007.

[18] J. G. De Jalon and E. Bayo. *Kinematic and dynamic simulation of multibody systems.* Springer-Verlag New York, USA, 1994.

[19] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[20] P. E. Duda and R. O. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, Inc., New York, New York, 1973.

[21] M. Eicholtz and L. B. Kara. Intermodal image-based recognition of planar kinematic mechanisms. *Journal of Visual Languages & Computing*, 27:38–48, 2015.

[22] M. Eicholtz, L. B. Kara, and J. Lohn. Recognizing planar kinematic mechanisms from a single image using evolutionary computation. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 1103–1110, New York, NY, USA, 2014. ACM.

[23] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[24] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.

[25] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *Int J Comput Vis*, 61(1):55–79, 2005.

[26] M. A. Fischler and R. Elschlager. The representation and matching of pictorial structures. *Computers, IEEE Transactions on*, C-22(1):67–92, Jan 1973.

[27] L. Fu and L. B. Kara. Recognizing network-like hand-drawn sketches: a convolutional neural network approach. In *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 671–681. American Society of Mechanical Engineers, 2009.

[28] L. Fu and L. B. Kara. From engineering diagrams to engineering models:

Visual recognition and applications. *Computer-Aided Design*, 43(3):278–292, 2011.

[29] L. M. Fugikawa, J. R. Morgan, F. E. Shelton IV, and E. L. Timperman. Pneumatically powered surgical cutting and fastening instrument with mechanical linkage coupling end effector and trigger motion, Oct. 7 2008. US Patent 7,431,189.

[30] J. Ginsberg. *Engineering Dynamics*. Cambridge University Press, New York, New York, 2008.

[31] C. D. Godsil, G. Royle, and C. Godsil. *Algebraic graph theory*, volume 207. Springer New York, 2001.

[32] W. W. Grist. Earth working apparatus, July 18 1978. US Patent 4,100,688.

[33] T. Hammond and R. Davis. Tahuti: A geometrical sketch recognition system for uml class diagrams. In *ACM SIGGRAPH 2006 Courses*, page 25. ACM, 2006.

[34] T. Hammond and B. Paulson. Recognizing sketched multistroke primitives. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 1(1):4, 2011.

[35] M. Hegarty. Mental animation: inferring motion from static displays of mechanical systems. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 18(5):1084, 1992.

[36] M. Hegarty. Mechanical reasoning by mental simulation. *Trends in cognitive sciences*, 8(6):280–285, 2004.

[37] M. Hegarty and V. K. Sims. Individual differences in mental animation during mechanical reasoning. *Memory & Cognition*, 22(4):411–430, 1994.

[38] C. H. Hines. Stretcher foot pedal mechanical linkage system, Feb. 9 1988. US Patent 4,723,808.

[39] L. B. Kara, L. Gennari, and T. F. Stahovich. A sketch-based tool for analyzing vibratory mechanical systems. *Journal of Mechanical Design*, 130(10):101101, 2008.

[40] L. B. Kara and T. F. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers & Graphics*, 29(4):501–517, 2005.

[41] M. Laribi, A. Mlika, L. Romdhane, and S. Zeghloul. A combined genetic algorithm–fuzzy logic method (ga–fl) in mechanisms synthesis. *Mechanism and Machine Theory*, 39(7):717–735, 2004.

[42] C. Lee, J. Jordan, T. F. Stahovich, and J. Herold. Newtons pen ii: An intelligent, sketch-based tutoring system and its sketch processing techniques. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, SBIM '12, pages 57–65, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[43] W. Lee, L. Burak Kara, and T. F. Stahovich. An efficient graph-based recognizer for hand-drawn symbols. *Computers & Graphics*, 31(4):554–567, 2007.

[44] H. Lipson. Evolutionary synthesis of kinematic mechanisms. *AI EDAM*, 22(3):195, 2008.

[45] Y. Liu and J. McPhee. Automated type synthesis of planar mechanisms using numeric optimization with genetic algorithms. *Journal of Mechanical Design*, 127(5):910–916, 2005.

[46] J. P. McCloskey. Portable trommel, Oct. 13 1998. US Patent 5,819,950.

[47] D. J. McGill and W. K. King. *Engineering Mechanics: An Introduction to Dynamics*. Tichenor Publishing, Bloomington, Indiana, 2003.

[48] J. L. Meriam and L. G. Kraige. *Engineering Mechanics, Volume 2*. John Wiley and Sons, Inc., Singapore, 1993.

[49] D. Mundo, J.-Y. Liu, and H.-S. Yan. Optimal synthesis of cam-linkage mechanisms for precise path generation. *Journal of Mechanical Design*, 128(6):1253–1260, 2006.

[50] T. Y. Ouyang and R. Davis. A visual approach to sketched symbol recognition. In *IJCAI*, volume 9, pages 1463–1468, 2009.

[51] S. Perera and N. Barnes. Maximal cliques based rigid body motion segmentation with a rgb-d camera. In *Computer Vision–ACCV 2012*, pages 120–133. Springer, 2013.

[52] E. J. Peterson, T. F. Stahovich, E. Doi, and C. Alvarado. Grouping strokes into shapes in hand-drawn diagrams. In *AAAI*, volume 10, page 14, 2010.

[53] A. Pothen and C.-J. Fan. Computing the block triangular form of a sparse matrix. *ACM Transactions on Mathematical Software (TOMS)*, 16(4):303–324, 1990.

[54] V. Ramakrishna, D. Munoz, M. Hebert, J. A. Bagnell, and Y. Sheikh. Pose machines: Articulated pose estimation via inference machines. In *Computer Vision–ECCV 2014*, pages 33–47. Springer, 2014.

[55] S. Ross, D. Munoz, M. Hebert, and J. A. Bagnell. Learning message-passing inference machines for structured prediction. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2737–2744. IEEE, 2011.

[56] G. Salton and M. J. McGill. Introduction to modern information retrieval. 1983.

[57] Y. Sato, J. Takamatsu, H. Kimura, and K. Ikeuchi. Recognition of a mechanical linkage based on occlusion-robust object tracking. In *Multisensor Fusion and Integration for Intelligent Systems, MFI2003. Proceedings of IEEE International Conference on*, pages 329–334. IEEE, 2003.

[58] T. Sessa. Car side window lifting mechanism, Mar. 4 1980. US Patent 4,191,060.

[59] P. Soille. Generalized geodesy via geodesic time. *Pattern Recognition Letters*, 15(12):1235–1240, 1994.

[60] E. Soylemez. *Mechanisms*. Middle East Technical University, Ankara, Turkey, 1993.

[61] N. Srinivas and K. Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248, 1994.

[62] Y. G. Woldesenbet, G. G. Yen, and B. G. Tessema. Constraint handling in multiobjective evolutionary optimization. *Evolutionary Computation, IEEE Transactions on*, 13(3):514–525, 2009.

[63] D. H. Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[64] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1385–1392. IEEE, 2011.

[65] W. Yoo and E. Haug. Dynamics of flexible mechanical systems using vibration and static correction modes. *Journal of Mechanical Design*, 108(3):315–322, 1986.

[66] A. L. Yuille. Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3(1):59–70, 1991.

[67] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *Evolutionary Computation, IEEE Transactions on*, 3(4):257–271, 1999.