# Security Analyses of Click-based Graphical Passwords via Image Point Memorability

Bin B. Zhu
Microsoft Research Asia
Beijing 100080, China
binzhu@microsoft.com

Jeff Yan
Newcastle University
NE1 7RU, United Kingdom
jeff.yan@ncl.ac.uk

Dongchen Wei*, Maowei Yang*
Sichuan University
Chengdu, Sichuan, China
dowei@microsoft.com,
djyangmaowei@gmail.com

## ABSTRACT
We propose a novel concept and a model of image point memorability (IPM) for analyzing click-based graphical passwords that have been studied extensively in both the security and HCI communities. In our model, each point in an image is associated with a numeric index that indicates the point's memorability level. This index can be approximated either by automatic computer vision algorithms or via human assistance. Using our model, we can rank-order image points by their relative memorability with a decent accuracy. We show that the IPM model has both defensive and offensive applications. On the one hand, we apply the model to generate high-quality graphical honeywords. This is the first work on honeywords for graphical passwords, whereas all previous methods are only for generating text honeywords and thus inapplicable. On the other hand, we use the IPM model to develop the first successful dictionary attacks on Persuasive Cued Click Points (PCCP), which is the state-of-the-art click-based graphical password scheme and robust to all prior dictionary attacks. We show that the probability distribution of PCCP passwords is seriously biased when it is examined with the lens of the IPM model. Although PCCP was designed to generate random passwords, its effective password space as we measured can be as small as 30.58 bits, which is substantially weaker than its theoretical and commonly believed strength (43 bits). The IPM model is applicable to all click-based graphical password schemes, and our analyses can be extended to other graphical passwords as well.

## Categories and Subject Descriptors
K.6.5 [**Management of Computing and Information Systems**]: Security and Protection – *Authentication, unauthorized access.*
K.4.4 [**Computers and Society**]: Electronic Commerce – *Security.*

## General Terms
Security, Experimentation.

## Keywords
Authentication, graphical honeywords, dictionary attacks, image point memorability.

* Dongchen Wei and Maowei Yang participated in this work as interns of Microsoft Research Asia.


*CCS'14*, November 3–7, 2014, Scottsdale, Arizona, USA.

http://dx.doi.org/10.1145/2660267.2660364

## 1. INTRODUCTION
A major development in computer security in the past decade is the emergence of "usable security", which has now become a thriving and fast-moving discipline. Most, if not all, people agree that we need security systems that are both secure and usable.

A hot research topic in usable security is graphical passwords, which aim to deliver a graphical alternative to ubiquitous text passwords that have long suffered from various security and usability problems. Graphical passwords have been extensively studied in both the security and HCI communities, and have seen increasing deployments in the real world. For example, Microsoft Windows 8 has adopted a variant of the BDAS graphical password scheme [1]. Android's unlocking pattern scheme is also a graphical password system.

A major inspiration for graphical password research is the well-known fact that people perform far better when remembering pictures rather than words. As the saying goes, a picture is worth a thousand words. This picture superiority effect has been studied for decades by cognitive scientists and psychologists [2,3]. However, these studies have mainly focused on the memorability of pictures or images as a whole.

A notable exception is an MIT team's pioneering work published in 2012 [26]. They introduced the notion of "memorability of image regions", and studied which regions of an image are memorable or forgettable, with the purpose of using individual regions to predicate the whole image's memorability.

In this paper, we study memorability of image points, in the aim of making breakthrough in security analyses of click-based graphical passwords, in which a password is a sequence of points, referred to as *click-points*, on one or more images. Such graphical passwords include a family of schemes, and have been studied the most extensively. We propose a novel model to capture and describe the memorability of each clickable point in an image. In our definition, a *point* is the smallest measure of an image. It has no internal structure. In contrast, a *region*, as studied in [26], is a portion of an image that has internal structures and comprises many image points. Additionally, a *spot* is a very small portion of an image that comprises points considered equivalent in click-based graphical passwords. For example, points within a tolerance distance to a click-point are acceptably equivalent to the click-point in verifying a password. These points form a spot. In this paper, when the context makes it clear, we may use a point to refer to the spot comprising its equivalent points, and reversely use a point's spot to refer to the point.

Our work was inspired by fine prior work [12-18], which shows that image points are not selected with equal chances: people prefer some points to others in creating passwords. An implied interpretation is that people associate different memorability levels

to different points. However, we go several steps forward from the state of the art. Our main contributions are as follows.

For the first time, we formulate the concept of image point memorability (Section 3). We model the determination of a point's memorability as a noisy process in human memory where both memorable and forgettable aspects (i.e. both memory retention and memory decay) matter. Accordingly, for the first time, we have developed a heuristic model to accommodate numerical methods for describing, measuring, and comparing memorability levels of image points.

As another novel contribution, we have developed two methods for implementing and utilizing the IPM model. One combines automatic algorithms with human efforts, and is called *human-assisted memorability* (Section 4). The other is a computational approximation using automatic computer vision algorithms, and is called *automated memorability* (Section 5). While some of our vision algorithms are similar to those in [13,16-18], we have developed new algorithms by incorporating available vision techniques to either improve those used in the prior art or realize new functionalities unavailable. This computational realization of our IMP model is not straightforward, but requires significant, novel and creative efforts.

We will show for the first time that our IPM model has both defensive and offensive applications in security.

On the defensive side, the IMP model can be used to generate high-quality honeywords for any click-based graphical password schemes (Section 6). Honeywords (false passwords) were introduced at ACM CCS'13 by Juels and Rivest [20] for detecting password compromise, but they only introduced methods for generating honeywords for text passwords. How to generate honeywords for graphical passwords remains an open question. In this paper, we will offer the first methodology and the first experimental study of honeyword generation for graphical passwords. We will also offer a set of criteria for judging honeyword quality, and show that honeywords generated with our methods are of a high quality.

On the offensive side, the IPM model offers a new approach to mounting effective dictionary attacks on click-based graphical passwords (Section 7). We develop the first successful dictionary attacks on Persuasive Cued Click Points (PCCP) [7,8], which is the state-of-the-art click-based scheme and robust to all prior dictionary attacks. We show for the first time that, when examined with the lens of our IPM model, the probability distribution of PCCP passwords is seriously biased. Although PCCP was designed to generate random passwords, its security is substantially weaker than previous common beliefs.

Our contributions also include discussions of practical relevance of our attacks, and measures that we recommend for improving the practical security of PCCP.

# 2. RELATED WORK

## 2.1 Click-based Graphical Passwords

### *2.1.1 Representative Schemes*

While the first envisaged graphical password scheme [22] is click-based, the most extensively studied click-based graphical password scheme is PassPoints [9,10], wherein a user selects a sequence of $r$ click-points anywhere on an image in creating a password, and re-clicks the same sequence of click-points within a preset tolerance range in authentication. Previous studies [9-11] suggested that $r = 5$ results in a good balance of security and usability. This configuration has thus been widely adopted in studying click-based graphical passwords.

As described later, PassPoints is vulnerable to attacks exploiting image hotspots [12,13] and click patterns [14]. To address these vulnerabilities, Cued Click Points (CCP) [19] extends PassPoints by using multiple images, one click-point per image, for a password, and the next image is determined by a deterministic function of the current image, the clicked tolerance square, and the user ID. Persuasive Cued Click Points (PCCP) [7,8] extends CCP by requiring a user to select a click-point within a randomly positioned viewport in creating a password. A "shuffle" button is provided for users to randomly reposition the viewport until a click-point is selected.

### *2.1.2 Dictionary Attacks*

Security of click-based graphical passwords has been extensively studied. Golofit [12] examined the relationship between user-created PassPoints passwords and the features of images used, and found that users tended to avoid flat regions, irregular structures, and periodic regular structures in selecting click-points. Dirik et al. [13] also found that the click-points of user-created PassPoints passwords tend to concentrate at certain spots, i.e., image hotspots. Chiasson et al. [14] found that there exist distinct common patterns among click-points of a PassPoints password, i.e., click patterns.

Hotspots and click patterns have been exploited to mount successful dictionary attacks on PassPoints. For example, Dirik et al. [13] proposed an attack on PassPoints. Assuming that people tend to choose as click-points the centroids that draw natural visual attention in creating passwords, this attack uses a visual attention model to predict the likelihood of a centroid to be a click-point. It has achieved a success rate of 8.45% on a representative image using a dictionary of about 31 bits (i.e., $2^{31}$ entries), whereas the theoretical password space was 40 bits [13].

Automated dictionary attacks on PassPoints were also proposed in [16,17], while the most comprehensive and sophisticated automated dictionary attacks on PassPoints were reported in [18]. In these attacks, both corners and centroids are detected as candidates of click-points. Heuristic click patterns and salient regions detected with a visual attention model are then used to select likely combinations of click-point candidates in building attack dictionaries. On two representative images with a theoretical password space of 43 bits, van Oorschot et al. [18] achieved a success rate of 7-16% using dictionaries of approximately 26 bits built with a simple pattern combined with the visual attention model, and a success rate of 48-54%, the highest success rate ever reported, using dictionaries approximately 35 bits built with image-independent patterns.

Human-seeded attacks on PassPoints were reported in [15,16]. In these attacks, click-points from a small set of users were harvested for targeted images, and attack dictionaries are constructed using either a first-order Markov model or an independent probability model. On two representative images with a theoretical password space of 43 bits, the human-seeded attacks achieved a success rate of 20-36% using dictionaries of 31 to 33 bits built with the independent probability model, and a success rate of 4-10% within 100 guesses using the first-order Markov model – the most efficient dictionary attack ever reported.

Chiasson et al. [14] examined hotspots and click patterns for PassPoints, CCP, and PCCP. The study concludes that PassPoints passwords contain both hotspots and click patterns, CCP passwords contain hotspots but no click patterns, and PCCP passwords contain

neither hotspots nor click patterns. A later study [8] reexamined the data collected in [14] and in other studies, including two PCCP studies wherein the PCCP passwords were created with much more shuffles than those in [14]. According to [8], the click-points of the PCCP passwords examined in [14] approach complete spatial randomness, whereas the click-points of all the PCCP passwords from the aforementioned three studies deviate from complete randomness. This is reasonable since more shuffles lead to less randomly distributed click-points. Using the click-point distribution of all the PCCP passwords, they estimated that a 33-bit dictionary had a mere 3% chance in successfully guessing a PCCP password in the set that traverses all possible combinations of the collected click-points [8]. It would be much more difficult to successfully guess an actual PCCP password since the actual PCCP passwords collected in those studies took only a tiny portion of the set. Therefore, harvesting click-points cannot attack PCCP. Harvesting real passwords directly cannot attack PCCP either, since PCCP passwords are user-dependent: with a high probability, each user gets a different sequence of images for her password. Therefore, PCCP is robust to human-seeded attacks that harvest either passwords or click-points.

An image typically has a large number of corners and centroids. Permuting all of them to form password candidates would lead to a huge dictionary that is ineffective for dictionary attacks. For PassPoints, this issue is resolved by applying click patterns and salience to select only combinations that are likely to be a password in building attack dictionaries. For PCCP, there is no pattern or correlation among click-points in a password at all. This leaves salience as the only means to reduce the number of candidates. Our study (see Section 7.5.1) suggests that the salience calculated with the state-of-the-art visual attention model does not predict click-points of PCCP passwords well. Therefore, prior automated attacks cannot build effective dictionaries to mount successful attacks on PCCP.

To summarize, PCCP is robust to all prior dictionary attacks. Such strength of PCCP is expected since it was designed to resist these attacks in the first place.

A click-point cannot be re-clicked exactly. Any click within a certain range of the click-point should be accepted as the click-point. This tolerable range is fine in verifying an input password if the original password is stored, but makes such verification impossible if only the hash value of a password is stored since different clicks result in different hash values. Password discretization [30,31] is designed to solve this problem. Our recent paper [25] found that representative discretization schemes leak significant password information. In the current paper, we do not exploit the weakness of password discretization at all.

The representative results of all prior attacks on click-based graphical passwords are summarized in Appendix.

## 2.2 Other Graphical Passwords

The graphical password schemes used in Windows 8 and Android belong to a different type that traces back to Draw-A-Secret (DAS) [21] wherein a user draws a password on a 2D grid, with the sequence of grid cells along the drawing path used as a password. Pass-Go [4] encodes grid intersection points rather than grid cells. A similar scheme is used in Android to unlock phones, which was recently attacked [6]. Background Draw-A-Secret (BDAS) [1] adds a background image to DAS to provide cue for re-drawing a password. The Windows 8's variant allows users to draw a combination of lines, circles, and taps on an image as a password. This scheme was also attacked recently [5].

More information on graphical passwords can be found in good survey papers [23,24].

# 3. MEMORABILITY OF IMAGE POINTS

## 3.1 Our Approach

Understanding the memorability of images as a whole or of individual image regions is relevant to the research of graphical passwords in general, but cannot contribute much to security analysis of click-based graphical passwords. We envisage that if each point on an image can be measured and compared in terms of memorability, this new approach will be at the right granularity for, and significantly contribute to, analyzing such graphical passwords.

The discovery of hotspots [12,13] and click patterns [14] suggests that click-points tend to be memorable points. Human-seeded and automatic methods [13,15-18] were explored to predict click-points in PassPoints with a good success. However, these fine prior art did not bring out the concept of building a model of image point memorability. Neither did they consider two important aspects of memorability, namely human memory decay and image content semantics.

Golofit [12] suggested that, when creating PassPoints passwords, users tended to avoid the three types of image regions shown in Figure 1. Our interpretation is simply that these regions do not have semantic concepts that are strong enough to resist human memory decay, and thus they are unmemorable and avoided by the users. The prior art [13,16-18] nicely ignores regions like Figure 1(A), but fails to realize that corners and centroids in Figures 1(B) and 1(C) are not memorable, for the exact reason that their methodology does not take into consideration both content semantics and memory decay.

Based on these prior art, we formulate the notion of image point memorability. Similar to [26], we consider both memorable and forgettable aspects, and model the memorability of an image point as a noisy memory process of mentally specifying the point on the image, and registering the information with human memory. In this process, at least two main factors matter, namely, the complexity of mentally specifying the point, which determines the memory burden for a user, and the point's distinctiveness in its surroundings, which reflects its robustness to memory decay. These are the main features differentiate our approach from all the prior art, methodologically.


**Figure 1: Three types of regions that users tend to avoid in creating PassPoints passwords (taken from [12]).**

## 3.2 A Heuristic IPM Model

We first define that an *object* is an identifiable portion of an image that can be interpreted semantically as a single unit. An object is typically an image region, but can degenerate into a point when the object is sufficiently small.

In our IPM model, the notions of both 'point' and 'object' are essential. As a key construct in our model, we assume that, when

people mentally specify an image point, this specification is in reference to an object. Three other key notions in our model – two for capturing memorable aspects of memorability, and the third for forgettable aspects – are as follows.

**Semantics** is important for image memorability [2,3] and relevant to both image points and objects. We use it to classify objects or points into three types:

- An object or point is of *high semantic coherence (HSC)*, if it represents a clear and unitary semantic concept, such as "dog" and "tree" for objects, and "center" and "corner" for points.
- An object or point is of *medium semantic coherence (MSC)*, if it is not a clear and unitary concept but is still semantically meaningful. Such an object or point has a clear "meaning" but is not as easily pinpointed as above. Instead, it can be pinpointed via a concise description, such as "edge of pool".
- An object or point is of *low semantic coherence (LSC)*, if it cannot be classified into either type above.

**Salience** refers to the state or quality of an object that stands out perceptually from its neighbors. A *salient* object naturally draws viewers' attention, but a non-salient object does not. Therefore, it is less complex to mentally specify a salient object than a non-salient one. In our model, the notion of salience is applicable only to image *objects*, but not to image points.

**Distinguishability** is a notion that is defined for both objects and points. We use it to classify a point or object into three types: *undetectable*, *detectable*, and *distinguishable*. A point or object is *detectable* if it can be visually identified on an image; otherwise *undetectable*. A detectable point or object is *distinguishable* if it is easily distinguished from the surrounding context. A detectable point may be indistinguishable. For example, a cross-point or a rectangle's center in Figure 1(C) is detectable but not distinguishable since it is hard for humans to mentally distinguish the point from surrounding similar points. To be robust to memory decay, a memorable point or object should be distinguishable. An indistinguishable point or object is unmemorable. Note that the term "distinguishable" carries a different meaning in [17,18], where distinguishable points are actually detectable points we refer to in this paper.

In a nutshell, our IMP model can be explained with the above notions as follows. The complexity of mentally specifying a point is measured with the point's and its reference object's semantics levels, and together with the reference object's salience level. A point's robustness to memory decay is measured by the distinguishability levels of both the point and its reference object. Intuitively, a typical memorable point is of HSC, related to a salient HSC reference object, and distinguishable from its surrounding neighbors.

We then develop a heuristic categorical system, which is so designed that will accommodate both automated and human-assisted realizations of our IMP model.

Indistinguishable points are unmemorable, and thus collectively treated as a single category in our model. On the other hand, by semantics, there are three types of distinguishable points: HSC, MSC and LSC. For these points, we have $2 \times 3 = 6$ types of reference objects, considering both salience and semantics. As a point is specified in reference to an object, this gives $3 \times 6 = 18$ possible combinations of distinguishable points and objects. For the sake of simplicity, we group them into 7 main categories, as explained below.

***Image Point Memorability (IPM) Model:***

- *We introduce M-index to measure a point's relative memorability reversely: a lower M-index value indicates better memorability.*
- *We assign an M-index of infinity to indistinguishable points, due to their poor memorability.*
- *A distinguishable point carries a weight describing its semantic coherence level, which is 0 (High), 1 (Medium) or 2 (Low); so does its reference object. In addition to a semantics weight, the object also carries a weight describing its salience level, which is either 0 (salient) or 2 (non-salient).*
- *A distinguishable point's M-index is the sum of its semantics weight and its reference object's semantics weight and salience weight.*
- *Heuristically, distinguishable image points and their relative memorability are classified into the following 7 categories according to their M-indices:*

| ***M-Index*** | ***Description*** |
|---|---|
| *0* | *HSC points of salient HSC objects.* |
| *1* | *MSC points of salient HSC objects, HSC points of salient MSC objects.* |
| … | … |
| *6* | *LSC points of non-salient LSC objects.* |

# 4. HUMAN-ASSISTED MEMORABILITY

The IPM model can be semi-automatically approximated by finding detectable points with computer vision algorithms and labelling detectable points by humans. This leads to our human-assisted memorability. We first describe a version independent of point locations, which fits PCCP-type graphical password schemes wherein the chance that a point is chosen as a click-point is independent of the point's location due to the randomly positioned viewport. Then we revise it to incorporate a point's location.

## 4.1 Detectable Points

Detectable points should be distinct structural points or points located in reference to distinct structural entity. Like [16-18], we approximate distinct structural points with corners and points located in reference to others with centroids. A *corner* is the intersection of two edges, and a *centroid* is the center of an object or its segment. However, the corner detection used in [16-18] tends to detect excessive corners, such as weak corners of small and irregular structures, which people unlikely select as click-points.

We address this problem by detecting as distinct structural points both weakly distinctive corners of large structures (i.e., long edges) and strongly distinctive corners. Our centroid detection is similar to [18]. The details of our corner and centroid detection algorithms are available in our recent paper [25]. Detected corners and centroids form a set of detectable points for an image, as shown in Figure 2 with a representative image. By excluding those points that are unlikely click-points, we produce a smaller yet more accurate set of detectable points than the prior art did.

**Figure 2: Detectable points.**

## 4.2 Location-Independent Human-Assisted Memorability

To facilitate labeling detectable points, we have implemented a tool to display detectable points on an image, with different categories marked with different colors. Detectable points are all initially assigned with M-index = 6. A user can move or adjust a circle to select detectable points inside the circle to label or adjust their M-indices according to the IPM model. We add a new category called *textual points* for detectable points that are a part of texture. A *texture* is a region comprising similar and repetitive structures, possibly with a certain degree of randomness. Both cross-points and rectangle's centers shown in Figure 1(C) are textual points. Textural points are unmemorable and thus removed.

Salient objects are first identified as follows: looking at the image far away such that image details are lost; objects that are still identifiable are considered as salient objects. Once salient objects are identified, the M-index of a point on a salient object is then reduced by 2, the weight for a salient object. Then HSC and MSC objects are identified. The M-index of a point on an identified object is reduced by the weight of the object, i.e., 2 or 1 for a HSC or MSC object. Finally, HSC and MSC points are identified, and their M-indices are reduced in a similar manner. The tool also allows a user to assign a point's M-index directly. Unfinished work can be saved and then resumed at a later time. Pressing "Submit" button ends labeling of an image.

We collected 1200 images in our empirical study of PCCP security (see Section 7). Three computer science undergraduates were hired to label detectable points in these images. They had not used any click-based graphical password before, and were otherwise not involved in this work. They were trained with exemplary samples of each category in advance. The task of labeling the 1200 images was divided among the three people. Each image was labeled by one person. The work was done in 10 days. The average time to label an image was 3 minutes and 32 seconds.

There was no cross-validation in this labeling process, which might result in some inconsistent or inaccurate labels. However, it turns out that the labelling quality we achieved was sufficient for our studies, as evidenced by experimental results we present later.

## 4.3 Human-Assisted Memorability

Salience is location-dependent: humans naturally look at objects near the center of an image [27]. This behavior is accounted for in [27] with a Gaussian blob centered at the image center, which is also applied to adjust human labelling results obtained in Section 4.2. The following empirical rule is applied: human labeled M-index is adjusted by subtracting the Gaussian blob normalized to an output range of [0, 1]. A detectable point at the image center is thus moved to the next category of lower M-index, while there is no change to the M-index of a detectable point at an image corner. The resulting M-index is shifted by adding 1 to ensure that M-index is non-negative.

The final M-index may be a non-integer, and can be further converted into 7 scales of the IPM model. However, it turns out that what matters for all applications discussed in this paper is the relative ranking capability enabled by these numeric values, for which the M-index without conversion serves equally well. As a result, such conversion is ignored in this paper.

# 5. AUTOMATED MEMORABILITY

The IPM model can also be approximated with automatic computer vision algorithms, albeit much coarser than the human-assisted approach since object recognition and image understanding remain open questions. This leads to our automated memorability. We will first describe the automated memorability, and then modify it to make it independent of a point's location.

## 5.1 Automated Memorability

The automated memorability comprises three modules. The first module, as described in Section 4.1, finds detectable points. For each detectable point, the second module applies the state-of-the-art attention model proposed by Judd et al. [27] to compute a salience value from 0 to 255, which predicts how likely, with a larger value meaning more likely, people would fixate at its local region.

The third module evaluates the point's distinguishability via dissimilarity of its neighborhood. The more dissimilar, the higher distinguishability. Colors and gradients are complement attributes of an image region, with gradients pertinent to structural information. Both color dissimilarity and structural dissimilarity are used. They indicate how dissimilar the color distribution and the gradient distribution, respectively, of a point's neighborhood.

The automated memorability computes the M-index of a detectable point empirically as follows:

$$M\text{-}index = \lambda_{color} \cdot \lambda_{grad} \cdot (255 - salience\ value), \qquad (1)$$

where "salience value" is calculated with the Judd et al.'s model, and $\lambda_{color}$ and $\lambda_{grad}$ are step-wise factors determined respectively by the color dissimilarity and gradient dissimilarity of the point. We describe in detail the determination of color dissimilarity and $\lambda_{color}$ in Appendix 11.2, and that of structural dissimilarity and $\lambda_{grad}$ in Appendix 11.3.

From Eq. (1), a detectable point with a higher salience value gets a smaller M-index (i.e., more memorable). Factor $\lambda_{color}$ (or $\lambda_{grad}$) is larger than 1 and thus boosts M-index (i.e., less memorable) if the point is in a less dissimilar region and thus less likely distinguishable from other points in its neighborhood. On the other hand, the factor is smaller than 1 and thus lowers M-index (i.e., more memorable) if the point is in a more dissimilar region and thus more likely distinguishable from other points in its neighborhood. For other types of local regions, its value is 1.

M-index given by Eq. (1) can be further converted into 7 scales of the IPM model. As we mentioned in Section 4.3, what matters for all applications discussed in this paper is the relative ranking capability enabled by these numeric values, for which M-index given by Eq. (1) serves equally well. As a result, we have ignored the conversion in this paper.

## 5.2 Location-Independent Automated Memorability

The automated memorability described above depends on a point's location since salience is location-dependent. However, we can eliminate this dependence simply by removing the center feature, the only location-dependent feature in the Judd et al.'s model [27]. The resulting M-index is the sought one.

## 5.3 Comparison with Prior Art

Prior art [13,16-18] has conceptually used the first two modules, but is inadequate for predicting memorability of a point due to lack of memory decay mechanism fulfilled in the third module, as we mentioned in Section 3.1. This inadequacy also manifests in the following real example. Figure 3 shows an image and its salience map, wherein a more salient area is brighter. We can see that leaves have rather large salience values. As Figure 2 shows, many detectable points can be found in leaves. Their M-indices given by Eq. (1) without using any dissimilarity factor will be small, meaning these detectable points are determined memorable. Actually they are textural points and thus unmemorable. The prior art is significantly inaccurate in evaluating the memorability of points in textual regions.

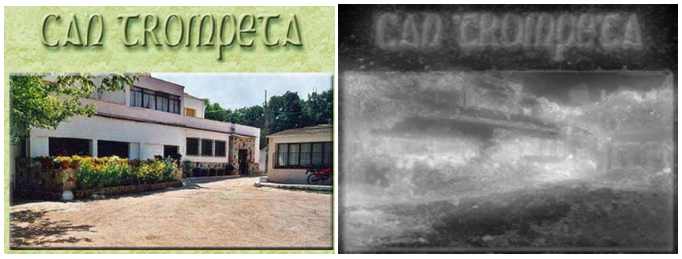

**Figure 3: Image (left) and its salience map (right).**

In contrast, by incorporating color and structural dissimilarity factors into Eq. (1) to model the memory decay mechanism in our IPM model, an M-index derived by the attention model is significantly boosted (i.e., much less memorable) for a point in leaves due to very low dissimilarity for both color and gradient distributions in leaves. Even though detectable points are still found in leaves, they are ranked among least memorable points (i.e., with largest M-indices), and thus are likely excluded from our dictionaries, which is what we seek for.

When a point's neighborhood is very dissimilar, the dissimilarity factors in Eq. (1) also decrease its M-index returned by the attention model (meaning the point is more memorable). In this way, relative memorability levels of detectable points are reordered by both color and structural dissimilarities, resulting in better and more accurate dictionaries than before.

In addition to our conceptual innovation of introducing a memory decay mechanism, we have developed a new corner detection algorithm that outperforms prior art, and have applied a better attention model than the one used in prior art.

Our approach offers the first automatic approximation to the IPM model, and distances itself from prior art, both conceptually and technically. We note that this is not necessarily the optimal automatic approximation. We encourage computer vision researchers to search for better techniques.

# 6. A DEFENSIVE APPLICATION: GRAPHICAL HONEYWORDS

Honeywords were recently introduced to improve the security of hashed text passwords [20]. For each account, one or more honeywords (false passwords) are cryptographically hashed and stored together with the real password hash. A legitimate user never uses the honeywords associated with her account. An adversary steals a file of hashed passwords, and can invert the hash function. If the adversary cannot tell passwords and honeywords apart, he may use a honeyword in a login attempt, which will sets off an alarm of password compromises. Honeywords are also applicable to systems that do not store password hashes, but store encrypted or plain passwords instead.

Graphical passwords are stored in a similar manner as text passwords. For click-based graphical passwords, a password breach may allow adversaries to access or deduce either a password's click-points or their tolerance squares, depending on whether password discretization has been used or not, and on which discretization scheme has been used. In the latter case, a tolerance square, which is very small, reflects the characteristics of the points inside. Graphical honeywords would serve exactly the same purpose for graphical passwords as text honeywords for text passwords.

The IPM model can be applied to generate honeywords for any click-based graphical password scheme. We chose PassPoints in our empirical study for two reasons:

- It is more challenging to design a honeyword scheme for PassPoints than for either CCP or PCCP, since the former exhibits click patterns, which the latter lacks but that can be exploited by adversaries to tell passwords and honeywords apart.
- To reach the same discriminative power, PassPoints requires a much smaller training set than CCP or PCCP does.

Thus, PassPoints facilitates a good feasibility study of our approach without loss of generality.

## 6.1 Honeywords: Desired Properties

To be effective and practical, a honeyword generation system should have the following desired properties:

1. *Efficiency.* Internet applications may have many accounts and thus need to generate a large number of honeywords efficiently.
2. *Indistinguishability.* Honeywords should be indistinguishable from passwords to prevent adversaries from telling them apart. This indistinguishability manifests in several ways:
   a. Honeywords should have the same semantic characteristics as passwords. For PassPoints, this means that honeywords should have a tendency to use hotspots and exhibit click patterns that PassPoints passwords exhibit.
   b. Passwords may exhibit certain variations due to human imprecision. For example, a hotspot is likely chosen as a click-point for PassPoints, but different passwords may contain a slightly different point due to human click variations. For a system that click-points can be deduced in password breaches, honeywords should also exhibit the same variations.
   c. Statistically indistinguishable. Honeywords should exhibit a probability distribution indistinguishable from that of passwords to avoid being distinguished by statistical analysis. For example, PassPoints passwords

containing hotspots and click patterns are more likely chosen by users than other passwords. Honeywords should also exhibit this skewed probability distribution.

Quality of generated honeywords can be evaluated by comparing with the ideal case that honeywords are truly indistinguishable from passwords. In this ideal case, the best one can do is a random guess, resulting in a success probability of $1/(N+1)$, where $N$ is the number of honeywords per password. This probability can be made arbitrarily small by selecting $N$ appropriately but at the cost of complexity and storage space. Without loss of generality, $N$ was set to 1 in our empirical studies.

## 6.2 Generating Honeywords for PassPoints

### *6.2.1 Overview*

In our honeyword generation, Property 2.a is achieved by utilizing the IPM model (Section 6.2.2) to select points and to rank-order honeyword candidates, referred to as *words*, and by adjusting the rank order with click patterns (Section 6.2.3) so that generated honeywords exhibit both hotspots and click patterns. To fulfill Property 2.c, each word is assigned a *drawing probability* according to which words are drawn (i.e., selected) in generating honeywords, and the drawing probability distribution is made similar to that of real passwords (Section 6.2.4) to avoid distinguishing honeywords and passwords statistically. We raise the challenge we face by requiring our honeyword generation to fulfill Property 2.b in order to be applicable regardless of password discretization methods. To achieve this, each drawn word is perturbed (Section 6.2.5) before outputting as a honeyword.

Either the automated memorability or the human-assisted memorability can be used as an approximation of the IPM model for generating honeywords. Either resulting system can generate a large number of honeywords efficiently. Thus Property 1 is fulfilled. Note that human labeling is needed for the human-assisted memorability, but such labeling is a one-shot, manageable effort. For example, it took less than 7 minutes to label the two images shown in Figure 4 that were used in our case study. After labeling, a large number of honeywords can be generated automatically and efficiently. We also note that honeyword generation with the human-assisted memorability is significantly different from generating honeywords with humans. It is a substantially easier task for a user to rank clickable image points by their memorability levels than to create a large number of distinct passwords as honeywords. More importantly, manually creating many honeywords is exhausting for a user, which deviates from people's normal password-creating behaviors. Honeywords created this way would highly likely have a distinct probability distribution from real passwords, resulting in honeywords distinguishable from passwords.

### *6.2.2 Building a Dictionary with IPM Model*

Detectable points are found from an image, and their M-indices are calculated using the IPM model to form a set of distinguishable points. For a simple image, this set might be too small. In this case, indistinguishable points of low M-indices (i.e., more memorable) are added to the set so that the set is large enough in order to generate a large variation of honeywords. On the other hand, when a simple image is used, security-conscious users may be forced to choose less memorable points in creating their passwords in order to avoid hotspots.

The points in the set are then traversed to form a dictionary of words; each word is a sequence of 5 distinct points with the minimum distance between any pair of points exceeding a threshold required by PassPoints. Like in attacking PCCP to be described in Section 7.1, each word is assigned an M-index that is the sum of the M-indices of its constituent points.

### *6.2.3 Adjustment by Click Patterns*

Since PassPoints passwords exhibit click patterns, so should honeywords. Two heuristic click patterns are considered: Line and Regular. *Line* includes any sequence of 5 click-points that follow a directional line. *Regular* includes any sequence of 5 click-points that follow consistent directions such as left to right, clock-wise.

Patterns are used to adjust M-indices of the words in the dictionary so that words exhibiting a pattern have a higher chance to be selected as honeywords, i.e., their M-indices are lowered. Like in Sections 5 and 6, this adjustment is achieved by multiplying a fractional factor for the automated memorability, and by subtracting a subtrahend for the human-assisted memorability. As described in detail in Appendix 11.4, this factor or subtrahend is a confidence level ranging from 0 to 1, which is an empirical metric to measure evaluation reliability that the word exhibits the pattern. For each word in the dictionary, Line is checked first. Regular is checked if the word does not exhibit the Line pattern. If an entry exhibits neither pattern, its M-index remains unchanged.

### *6.2.4 Approximating Password Distribution*

The drawing probability assigned to each word should preserve the word order by their M-indices. In addition, the drawing probability distribution should be similar to the probability distribution of passwords so that honeywords cannot be distinguished statistically from passwords. Since no probability distribution of PassPoints passwords is available, we approximate it with the distribution of attacking PassPoints, shown in Figure 6 in Section 7.5: the drawing probability of a word with M-index $C$ is calculated empirically as follows:

$$P(C) = \alpha \cdot \exp(-\beta \cdot \frac{C-Min}{Max-Min}), \qquad (2)$$

where $Min$ and $Max$ are the minimum and maximum M-index of the words in the dictionary, respectively, $\alpha$ is a normalization factor, and $\beta \geq 0$ is a parameter. Eq. (2) is a monotonically decreasing function of M-index, thus preserves the M-index order of words.

To mitigate variations due to different images, parameter $\beta$ is determined by matching the ratio of two points on the curve near each end. The curve of the automated attacks on PassPoints shown in Figure 6 is used for the dictionary generated with the automated memorability, and that of the human-seeded attacks is used for the dictionary generated with the human-assisted memorability.

### *6.2.5 Generating Honeywords*

When a honeyword is wanted, a word is drawn from the words in the dictionary according to their drawing probabilities. This can be realized by generating a random number with a uniform distribution from 0 to 1, and selecting the word that contains the random number in the cumulative probability. Each point in the selected word is then perturbed using the distribution of human click-variations given in [19] to fulfill Property 2.b. The result is output as a honeyword.

## 6.3 Empirical Evaluations

### *6.3.1 Experimental Setting*

We apply a simple yet powerful method to evaluate the quality of generated honeywords: using human brains in a standard learning test. In this method, experts learn from a training set of labelled passwords and honeywords. Then they are asked to identify each

password in a testing set disjoint with the training set in a blind test wherein the password and a honeyword are presented side by side randomly. The resulting *success rate*, which is the ratio of correctly identified passwords to total tested passwords, is then compared with the expected success rate of 50% of the ideal case that honeywords are indistinguishable from passwords.

Birds People

Figure 4. Two typical images in our experiments

Birds and People, the two images shown in Figure 4, were chosen in our case study since they represent two typical types of images possibly used in PassPoints: Birds is simple without many memorable points, while People is complex with many memorable points. Both images were 400x600. The passwords collected in a prior study [13], with 92 passwords for Birds and 142 passwords for People, were used in our experiments. Both automated memorability and human-assisted memorability were used to generate honeywords in order to compare their performances.

To facilitate quickly locating click-points in a password (or a honeyword) and viewing their surrounding contexts, three views were provided for a user to switch at will: the image overlaid with the password (i.e., the password's click-points and their order were marked on the image), the password alone, and the image alone. There was no restriction on how passwords in the training set were learned or how long to make a decision in a test.

Three experienced researchers in computer vision and pattern recognition who are familiar with graphical passwords acted as experts in our evaluation; otherwise they were not involved in this work. Before experiments, they were informed of the honeyword generation method. Their expertise enables them to understand our honeyword generation method inside out, and to know every clue and where and how to look for it in telling honeywords and passwords apart. Therefore they are the most capable adversaries in our setting.

For each expert, we randomly partitioned the passwords into three sets for each image: a training set with 30 passwords for Birds and 48 passwords for People, and two testing sets of equal size, one testing set for each realization of the IPM model. In the learning phase, we generated on the fly 100 honeywords per image per realization of the IPM model and labelled them with the realization method for each expert to learn.

In the testing phase, one password was tested at a time. First, we randomly selected a test set, and randomly picked an untested password from the set. Then a honeyword was generated from the IPM model realization corresponding to the selected test set. The password and the honeyword were displayed side by side at a random order. When a decision was made in a test, the correct answer was displayed so that finished tests could be learned for subsequent tests. This was to mimic the process that an adversary could learn from each trial. Our procedure design also eliminates the advantage that more samples were learned for a later tested realization if the two realizations were tested sequentially, resulting in a fair comparison of the two realizations of the IPM model.

At the end, each expert was asked to write down major discriminative features they had learned and employed in the tests.

Eight months after, we conducted another experiment to test a baseline method, wherein a honeyword was generated by randomly selecting 5 image points that satisfied PassPoints' requirements. By then, the three experts had already forgot all the passwords they saw before. This baseline experiment followed the same procedure as before, except that the results of the two test sets were averaged for each expert, since the honeywords used were all generated with the same baseline method.

### 6.3.2 *Experimental Results*

Table 1 shows the resulting success rates for each expert denoted as A, B, C as well as their averages. There is a significant difference for success rates between the baseline method and our methods. The baseline method's honeywords were almost perfectly identified, while our method's success rates are within a distance of 14.5% from the expected 50% success rate of the ideal case that honeywords are indistinguishable from passwords.

We also examined temporal behaviors of success rates, and did not find any noticeable improvement of success rates for late tests.

**Table 1. Experimental results: success rates (%)**

| % | Birds | | | People | | |
|---|---|---|---|---|---|---|
| | Auto | Human | Baseline | Auto | Human | Baseline |
| A | 64.5 | 48.4 | 98.4 | 63.8 | 44.7 | 100 |
| B | 58.1 | 51.6 | 96.8 | 61.7 | 48.9 | 98.9 |
| C | 45.2 | 58.1 | 100 | 55.3 | 42.6 | 100 |
| Average | 55.9 | 52.7 | 98.4 | 60.3 | 45.4 | 99.6 |

### 6.3.3 *Statistical Analysis*

Statistical analysis has been used to determine whether differences in data reflect actual differences or might reasonably have occurred by chance. A value of $p < 0.05$ is regarded as indicating statistical significance, implying less than 5% probability that results occurred by chance.

First we applied the exact binomial test to individual experimental results to determine if the results reflected actual distinguishability between honeywords and passwords. The resulting $p$-values are shown in Table 2 except the last row, where statistically significant results are marked in yellow. No expert could detect a statistically significant difference between passwords and honeywords generated with either of our methods, whereas every expert could detect a statistically significant difference between passwords and honeywords generated with the baseline method.

**Table 2. $p$-values with the exact binomial test**

| $p$-value | Birds | | | People | | |
|---|---|---|---|---|---|---|
| | Auto | Human | Baseline | Auto | Human | Baseline |
| A | 0.150 | 1.000 | 1.54E-08 | 0.079 | 0.560 | 1.42E-14 |
| B | 0.473 | 1.000 | 2.98E-08 | 0.144 | 1.000 | 3.48E-13 |
| C | 0.720 | 0.473 | 9.31E-10 | 0.560 | 0.382 | 1.42E-14 |
| All | 0.300 | 0.679 | 4.51E-25 | 0.018 | 0.312 | 5.13E-41 |

Then we applied Fisher's exact test of independence to the experimental results to determine if there was any statistically significant difference among the experts. The resulting $p$-values for different images and methods are shown in Table 3. No statistically

significant difference is found among the experts. As a consequence, we pooled the three experts' results together and redid the exact binomial test. The resulting $p$-values are shown in the last row of Table 2. There is only one case that the statistical significance flips: now there is a statistically significant difference between passwords and honeywords generated with the automated memorability on image People. This can be explained by the fact that more samples lead to more power or higher sensitivity in the exact binomial test. Therefore the pooled data test could detect subtler difference that individual tests could not detect.

**Table 3. $p$-values with Fisher's exact test of independence**

| | Birds | | | People | | |
|---|---|---|---|---|---|---|
| | Auto | Human | Baseline | Auto | Human | Baseline |
| $p$-value | 0.344 | 0.811 | 0.774 | 0.747 | 0.869 | 1 |

The automated memorability's different distinguishability with these two images can be explained by one discriminative rule learned and employed by the experts: passwords tended to contain semantically meaningful and/or semantically correlated points. Lacking of semantics understanding, the automated memorability might generate honeywords semantically distinctive from passwords, which were likely identified with this discriminative rule.

Semantically meaningful points tend to be hotspots, particularly for an image with a limited number of such points. Security-conscious users tend to avoid selecting any hotspots in their passwords, and thus their passwords are less likely to contain semantically meaningful points for a simple image (which has a small set of such points) than for a complex image (which has a large set of such points). On the other hand, a distinguishable point is more likely to be a hotspot for a simple image with a small set of distinguishable points than a complex image with a large set of distinguishable points. As a result, the discriminative rule tends to be less effective for image Birds than for image People, since the former contains much fewer semantically meaningful or distinguishable points than the latter. This explains the automated memorability's different distinguishability for People and for Birds.

The human-assisted memorability, on the other hand, captures semantics well, and thus the above discriminative rule is not effective for the human-assisted memorability.

To sum up, our empirical study suggests that both realizations of the IPM model are substantially better than the baseline method. For both simple and complex images, honeywords generated with the baseline method are distinguishable from passwords, whereas those generated with the human-assisted memorability are not distinguishable from passwords. On the other hand, honeywords generated with the automated memorability are indistinguishable from passwords when a simple image is used, but distinguishable with a powerful and sensitive statistical test when a complex image is used.

# 7. AN OFFENSIVE APPLICATION: DICTIONARY ATTACKS

The IPM model can be applied to guess any click-based graphical passwords. We choose PCCP as our case study for two reasons:

- As discussed in Section 2.1.2, PCCP is the most secure click-based scheme, and thus the most challenging to attack.
- PCCP is the best in testing efficacy and power of an IPM model since memorability is the only exploitable attribute in guessing PCCP passwords, and PCCP has the most random click-points and thus the widest coverage of test points.

Like the security analysis of PCCP against offline attacks in [8], we assume that all server-side information is known. This means that attackers have access to all images, know the first image of a password and its each click-point's grid of tolerance squares, and can determine the next image from a guessed click-point on a current image and verify if a guessed password is correct or not. This assumption allows us to focus on the guessability of PCCP passwords and their effective space, and to compare PCCP's strength with other click-based passwords and with text passwords.

## 7.1 Building Personalized Attack Dictionaries

In PCCP, a next image is determined by a deterministic function of the user ID, the current image, and the clicked tolerance square. This results in different next images for different accounts even if the same point is clicked on an image. Therefore different accounts have different password spaces, but these password spaces have the same size. We need to build personalized dictionaries for each account in dictionary attacks on PCCP. This is very different from dictionary attacks on PassPoints, wherein a single attack dictionary is applicable for all accounts.

A PCCP password comprises 5 click-points. A password guess, referred to as a *word*, should also comprise of 5 points. Each word is assigned an M-index to measure its relative memorability. Since click-points in a PCCP password are independent of each other, the M-index of a word is empirically the sum of the M-indices of its constituent points.

For each account, a personalized dictionary is built recursively by guessing one click-point at a time. More specifically, we start by guessing the first click-point of the password. Distinguishable points of the first image are found and grouped into different tolerance squares. If there is more than one distinguishable point in a tolerance square, the one with the smallest M-index (i.e., most memorable) is kept while the others are deleted. Each survived distinguishable point is a guess of the first click-point. They form a dictionary $D^1$ of words; each word contains one point.

Then we guess the second click-point. For every word $g_i^1$ in dictionary $D^1$ obtained above, $g_i^1 \in D^1$, we get the next image for $g_i^1$, find and process distinguishable points of the image as described for the first image. Each survived distinguishable point is a guess of the second click-point and added to $g_i^1$, resulting in a word containing 2 points, which is added to a new dictionary $D^2$ for the current stage. After processing all the words in $D^1$, we obtain a dictionary $D^2$ of words; each word contains 2 points.

The above process is repeated until all the five click-points have been guessed, resulting in a dictionary $D^5$. M-index is then calculated for each word in $D^5$, and all the words in $D^5$ are rank-ordered from low to high M-index (i.e., the most memorable word first). The sorted dictionary, possibly truncated to a specific size, is the sought attack dictionary for the account.

## 7.2 Experimental Setting

PCCP was implemented for Web applications according to [7,19]. A total of 1200 images with a similar structural complexity as those used in [15-18] were collected from the Internet, including the two images used in [18]. They were cropped, if necessary, to $451 \times 331$, the same size as in [15-18]. Hash function MD5 was used as the deterministic function to select the next image: the MD5 hash value was divided by 1200, the total number of images; the image

indexed by the remainder was selected. The viewport size was set to $75 \times 75$, the same as in [7,8].

Like [18], we used the centered discretization proposed in [30] and the tolerance square size $19 \times 19$, that is, a tolerance of 9 pixels along both horizontal and vertical directions from a click-point. Passwords were of 5 click-points, resulting in a theoretical password space of about 43 bits for each account: an image contains $[451/19] \times [331/19] = 391$ grid-squares; there are 391 possibilities for each click-point, and $391^5 \approx 2^{43}$ possibilities or 43 bits for five click-points. Note that this is the same size as the theoretical password space of PassPoints when it is implemented with the same configuration.

## 7.3 Password Collection

We recruited 96 experienced computer users, who were high school or college students, or staff members. They included 51 males and 45 females, and their age ranges from 16 to 48. Before password collection, they were trained for the tasks to perform. Each used a Web browser on his/her computer to create a password with and log into a remote authentication server.

Each participant was asked to create a password of 5 click-points for a fictitious bank account. The participants were encouraged to shuffle less to create more random and thus more secure passwords. A created password was confirmed immediately. After password enrollment, each participant was required to pass two login tests, namely 1-day and 7-day recalls. Each test allowed up to three trials. A participant who failed either test was required to re-create a password and to repeat the above procedure. Only passwords successfully passing both tests were collected. This was to ensure that collected passwords were all memorable for a reasonable time. We collected 96 passwords, one from each participant. They were created with on average 0.45 shuffles per click-point and a maximum of 2 shuffles per click-point.

## 7.4 Experimental Results

For each participant, we built personalized dictionaries of 35 bits (i.e. $2^{35}$ entries) as described in Section 7.1. Words in a personalized dictionary were tested one by one in the order of their M-indices from low to high until either the password was found (a success) or the dictionary was exhausted (a failure).

Figure 5 shows the number of passwords found at *n*-th bit range of guesses (i.e., found at the order of $2^n$ guesses but not at or before the order of $2^{n-1}$ guesses) for dictionaries built with the automated memorability and with the human-assisted memorability. The human-assisted memorability found passwords with much fewer guesses than the automated memorability. For example, the former found 2 passwords at the 25th bit range of guesses, and one more password at the 26th bit range of guesses, whereas the latter did not find any passwords until at the 32nd bit range of guesses. Figure 5 also shows success rates with dictionaries of different sizes. Note that our 35-bit dictionaries could be truncated to any smaller size since dictionary entries were fully ordered.

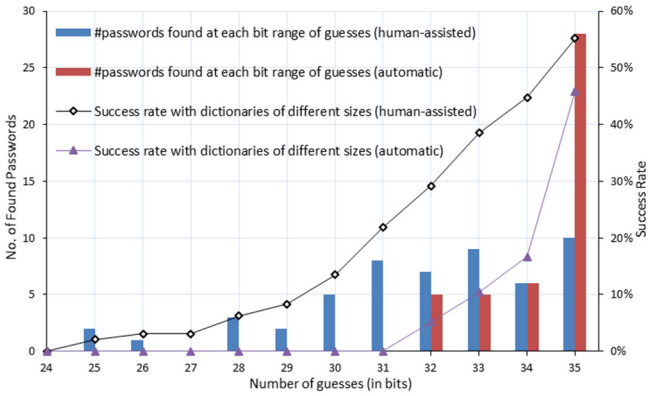


**Figure 5: No. of found passwords at each bit range of guesses and success rate with dictionaries of different sizes.**

## 7.5 Analyses and Comparison

To evaluate the attack efficiency, we adopt the partial guessing metric $\alpha$-work-factor $\mu_\alpha$ recently introduced in [32], which measures the fixed minimum number of guesses per account needed to achieve success rate $\alpha$. This metric can be converted to an effective key-length, denoted by $\tilde{\mu}_\alpha$, which is the size of a space of uniformly distributed passwords that would produce the same value of $\alpha$-work-factor [32], and defined in this paper as the *size of the effective password space (SEPS)* at success rate $\alpha$.

SEPS $\tilde{\mu}_\alpha$ has following properties. If an attack has a correct guess of the probability distribution of passwords and tests guesses in the order of their probabilities from high to low, then $\tilde{\mu}_\alpha$ estimated from attack results increases with $\alpha$: $\tilde{\mu}_\alpha \leq \tilde{\mu}_{\alpha+\epsilon}$, where $\epsilon > 0$. For a random guess attack, $\tilde{\mu}_\alpha$ estimated from attack results is independent of $\alpha$, i.e., a constant with respect to $\alpha$. In addition, if the probability distribution of passwords is a uniform distribution, $\tilde{\mu}_\alpha$, calculated from either attack results or the password probability distribution, is always independent of $\alpha$.

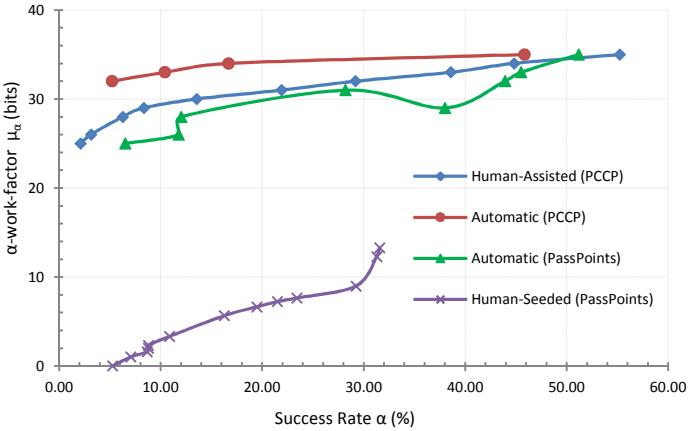


**Figure 6: Attack efficiency: number of guesses.**

Figure 6 shows attack efficiency of $\alpha$-work-factor $\mu_\alpha$ vs. success rate $\alpha$ for both the automated memorability (in red circle) and the human-assisted memorability (in blue diamond). It also shows attack efficiency of two most efficient attacks on PassPoints: the automated attacks reported in [18] (in green triangle) and the human-seeded attacks reported in [15,16] (in purple cross), wherein the average of success rates on the two representative images was used as the success rate, and the highest success rate was used if multiple success rates were reported for a given bit range of

guesses. Figure 7 shows sizes of effective password spaces at different success rates for each attack shown in Figure 6. Note that the results of PCCP and PassPoints shown in both figures were obtained with the same setting. Their theoretical password spaces were all 43 bits, and thus they are comparable.

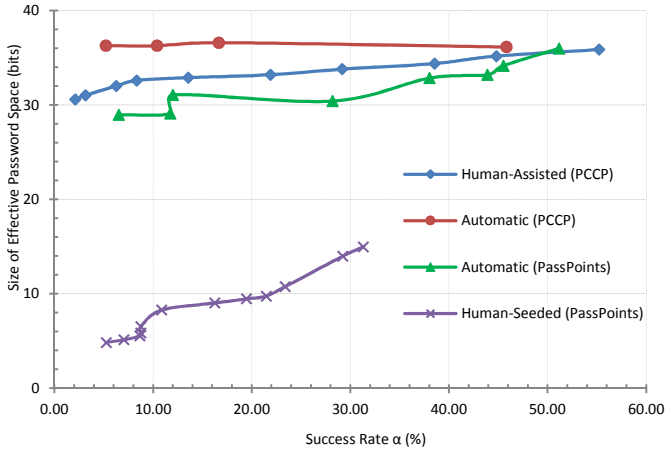


**Figure 7: Attack efficiency: size of effective password space.**

### *7.5.1 Analyses of PCCP Attack Results*

With 35-bit dictionaries, the success rate is 55.21% for the human-assisted memorability and 45.83% for the automated memorability, and their effective password spaces are 35.86 and 36.13 bits, respectively.

When success rate $\alpha$ decreases, the effective password space remains nearly a constant size, varying within the range from 36.13 to 36.58 bits, with an average of 36.31 bits, for the automated memorability. However, the effective password space reduces from 35.86 bit at $\alpha = 55.21\%$ to 30.58 bits at $\alpha = 2.08\%$ for the human-assisted memorability. Meanwhile, the attack efficiency with respect to a random guess attack remains nearly at a constant improvement of $2^{43-36.31} = 103$ times for the automated memorability and improves from $2^{43-35.86} = 141$ to $2^{43-30.58} = 5480$ times for the human-assisted memorability. The latter is substantially more efficient than the former.

The human-assisted memorability attack results imply that the PCCP passwords were not uniformly distributed. The trial order of words in this attack was positively correlated to the probability of PCCP passwords: passwords of higher probabilities were likely tested before passwords of lower probabilities. This indicates efficacy of our memorability model.

The automated memorability attack results, on the other hand, suggest that words in a 35-bit dictionary had the same probability to be the password – this is equivalent to random guesses. That is to say, the M-index ranking order calculated with the automated memorability is irrelevant to the probability distribution of passwords. This is unsurprising since the automated memorability approximates the IPM model with only salience and distinguishability, but largely ignoring semantics of both points and objects.

The automated memorability attack results also suggest that salience alone predicts click-points of PCCP passwords poorly. This can be explained by the fact that salience has a granularity of image region, which may not be closely correlated to which point in the viewport people likely select as a click-point in creating a PCCP password.

We conclude that the automated memorability is effective in eliminating indistinguishable points (i.e., points unlikely to be a click-point) but ineffective in predicting click-points among distinguishable points.

### *7.5.2 Comparison with Prior Analyses of PCCP*

A previous study [8] has examined PCCP's security by studying the spatial distribution of click-points in PCCP. It estimates that a 33-bit dictionary has a mere 3% chance in successfully guessing a password in the set that traverses all possible combinations of the collected click-points. The actual passwords take a tiny portion of the set, and thus this estimated strength should be much weaker than the actual strength of PCCP passwords. However, their approach and finding can be used to estimate the effective space for our PCCP passwords.

According to [8], the click-points of the PCCP passwords in [14], which were created with a mean of 3 shuffles and a median of 1 shuffle per click-point, approach complete spatial randomness. Using the same configuration, our passwords were created with fewer shuffles and thus should be more random than those in [14]. This suggests that the click-points of our PCCP passwords should also approach complete spatial randomness, which implies that the effective space of our PCCP passwords approaches the theoretical size, i.e., 43 bits.

Our attack results indicate that the strength of our PCCP passwords is substantially weaker than the above estimate. Our automated memorability attack suggests that their effective space is about 36 bits; our human-assisted memorability attack suggests that their effective space can be as small as 30.58 bits, and that their probability distribution is seriously biased.

Therefore, both the automated memorability attack and human-assisted memorability attack provide a substantially more accurate estimation of the strength of PCCP passwords than the previous study [8]. Our attacks are the first successful dictionary attacks on PCCP, which is robust to all prior dictionary attacks.

### *7.5.3 Comparison with Attacks on PassPoints*

Figures 6 and 7 show that the human-seeded attacks on PassPoints are much more efficient than other attacks on either PassPoints or PCCP. When success rate $\alpha$ decreases from 31.6% to 7.0%, the effective PassPoints password space reduces from 15.0 bits to 4.8 bits. This is dramatically smaller than the theoretical password space (43 bits). This implies that the probability distribution of PassPoints passwords is highly skewed: a significant portion of passwords have much higher probabilities and thus are highly predictable. The probability distribution of PCCP passwords, on the other hand, is much less skewed: the effective password space remains 30.58 bits at $\alpha = 2.08\%$ while the theoretical password space has the same size. These results indicate that PCCP passwords are much harder to predict and thus much more resilient to dictionary attacks than PassPoints passwords, which agrees with our expectation that the viewport in PCCP would result in more random passwords, and explains why prior dictionary attacks, while effective in attacking PassPoints, are ineffective in attacking PCCP.

Figures 6 and 7 also show that the automated attacks on PassPoints are slightly more efficient than our attacks on PCCP, albeit PassPoints passwords are much more predictable. With 35-bit dictionaries, they have similar success rates and similar sizes of effective password spaces, all around 50% and 36 bits, respectively. When success rate $\alpha$ decreases from around 50%, their effective password spaces start to deviate in size: the automated attacks on

PassPoints have a smaller size than our attacks on PCCP; the largest difference is about 3.5 bits for our human-assisted memorability attack and about 7.2 bits for our automated memorability attack. The large gap between the two curves of PassPoints attacks in Figure 7 indicates that the automated attacks on PassPoints did not capture the password probability distribution well enough, partially due to their lack of memory decay mechanism, i.e., distinguishability described in Section 5.1, in building attack dictionaries.

## 7.6 PCCP Password Distribution Bias

Here, we attempt to explain root causes of the PCCP password bias identified by our attacks, and why our IPM model can exploit the bias to predict PCCP passwords.

Firstly, choosing a PCCP click-point is effectively equivalent to choosing a PassPoints click-point from the image portion covered by the viewport, referred to as the *selection region* of the click-point. A selection region contains multiple spots since the $75\times 75$ viewport is much larger than a $19 \times 19$ tolerance square. Among these spots, it is not surprising that some spots are more popular than others. That is to say, the hotspot effect may exhibit in the selection region of a click-point, although such *local hotspots* are not necessarily hotspots for the whole image. As local hotspots are more likely chosen than other spots in the selection region, this results in a biased distribution.

Secondly, the distance between two points affects the chance that they fall into a selection region. If the viewport is never shuffled, two distant points whose spots would never fall into the same viewport area are independent in likelihood to be selected as a PCCP click-point. They do not contribute any bias to the password probability distribution. Sorting them by memorability levels is irrelevant and thus would not help pinpoint the click-point. In this case, the probability distribution bias is contributed by the preference for selecting as a click-point a "local hotspot" rather than any nearby points that are likely in the viewport area. This is the main cause to the distribution bias in our case.

Since a local hotspot is likely more memorable, rank-ordering all points that are likely in the same viewport area by their memorability levels helps pinpoint a click-point and thus detect the biased password distribution, as our attacks do. However, such rank-ordering of distant points that would never fall into the same viewport area does not help attackers at all, which explains our earlier observation in Section 7.5.3 that PCCP passwords are substantially more difficult to guess than PassPoints passwords.

However, when the viewport is shuffled in selecting a PCCP click-point, distant points that would never fall into the same viewport area may fall into the selection region of a click-point, and thus are no longer independent. Shuffling breaks their independence, and one point may be more likely selected as a click-point than the other. In this case, shuffling is another source contributing to the probability distribution bias, and rank-ordering these distant points by their memorability levels may also help pinpoint click-points and thus detect the bias.

## 7.7 Practical Implications

As we have found, the effective PCCP password space can be as low as 30.58 bits. This is substantially weaker than PCCP's theoretical strength and its commonly believed strength. To put it in perspective, this effective space size is about the same as the theoretical password space of Windows 8's graphical scheme (which is 30.1 bits according to [5]), significantly larger than the effective password space of typical text passwords (which is about middle twenties or fewer bits [32]), but less than the effective password space of strong text passwords used for high-value accounts (which is about 35 to 45 bits [33]). Therefore PCCP still offers reasonable security.

To improve the practical security of PCCP, we recommend increasing password length. Our experiment suggests that each click-point contributes about 6.12 (=30.58/5) bits in PCCP. If an effective password space of $M$ bits is needed, the number of click-points in a password should be $\lceil M/6.12 \rceil$. For example, if the password length is set to 8 click-points, still a click-point per image, the effective password space will be boosted to about 49 bits, which outperforms what strong text passwords used for high-value accounts can offer. However, a usability study is needed to gauge how well users can cope with the increased length in terms of password memorability.

We also recommend increasing image size. The images used in previous studies and ours were relatively small, $451 \times 331$. If we increase the image size to $640 \times 480$, the theoretical space of 5-click passwords will increase from 43.1 bits to 48.4 bits. If we use images that have a similar density of memorable points as in our case study, the effective password space will increase from about 30.58 bits to about 36 bits. We expect such a change in image size will have little or no negative impact on password usability.

Finally, it improves PCCP security by reducing the number of shuffles allowed in choosing a click-point during password creation, and by reducing the viewport size. These measures will increase randomness in passwords. However, both could introduce usability concerns, and therefore careful empirical studies are needed for determining an appropriate design choice.

# 8. CONCLUSION

For the first time, we have introduced the concept of image point memorability, proposed the first heuristic IPM model and two methods to implement it, and applied them to novel and successful security analyses of click-based graphical passwords, leading to both new techniques and insights.

On the defensive side, the IMP model can be used to generate high-quality honeywords for any click-based graphical password schemes. Our contribution in this arena includes the first systematic method for generating high-quality graphical honeywords, and a set of criteria for judging honeyword quality. Our empirical study with PassPoints indicates that our method could generate honeywords that exhibit no statistically significant difference from real passwords. The honeywords generated with either realization of our IPM model were substantially better than a baseline method that generates honeywords by randomly selecting image points.

On the offensive side, the IPM model offers a new approach to mounting effective dictionary attacks on any click-based graphical password schemes. In this arena, we have presented the first successful dictionary attack on PCCP, which is robust to all prior dictionary attacks. Our empirical evaluation of PCCP security via the IPM model shows that the probability distribution of PCCP passwords was seriously biased, in contrast to previous common beliefs. In our empirical study, the effective PCCP password space can be as small as 30.58 bits, which is substantially weaker than both its theoretical strength (43 bits) and its commonly believed strength. This result is a major contribution to the understanding of the real-world security of click-based passwords, and has significant implications.

To improve the practical security of PCCP, we recommend increasing its password length and image size. We also recommend considering the option of reducing the number of shuffles allowed in choosing a click-point during password creation, and the option of reducing the viewport size. As some of these measures might introduce usability concerns, we also recommend careful empirical studies to determine the right design choices for PCCP.

The power of our IPM model, as evidenced by the achievements, stems mainly from 1) the insight that both memorable and forgettable aspects of image points should be captured, and 2) a memorability metric that is defined at the right granularity and that supports appropriate numerical methods for describing, measuring, and comparing relative memorability ranking orders of image points. All these also represent our main conceptual and technical innovations that distance this work from prior art.

Our work sheds light on a new direction for analyzing graphical passwords, with ample potential for future research. For example, it is interesting to explore the IPM model's applicability in analyzing security of, and in generating honeywords for, other graphical passwords. It is also useful to develop refined computer vision algorithms for approximating image point memorability. These are our ongoing work.

Last but not the least, our work on image point memorability complements the research on the memorability of images as a whole and the memorability of individual image regions in multiple disciplines. We expect that it will interest a wide range of communities, including computer security, HCI, computer vision, psychology and cognitive science.

## 9. ACKNOWLEDGMENTS

We thank Ahmet E. Dirik for sharing the PassPoints passwords used in [13], Jussi Palomaki and Beibei Liu (both at Newcastle University, England), Alexei Czeskis and the anonymous reviewers for valuable comments, and our experiment helpers and participants for their contributions.

## 10. REFERENCES

[1] Dunphy, P., and Yan, J. 2007. Do background images improve "Draw a Secret" graphical passwords? In *ACM CCS'07*.

[2] Paivio, A., Rogers, T.B., and Smythe, P.C. 1968. Why are pictures easier to recall than words? *Psychonomic Science*. 11, 4, 137-138.

[3] Defeyter, M.A., Russo, R., and McPartlin, P.L. 2009. The picture superiority effect in recognition memory: A developmental study using the response signal procedure. *Cognitive Development*. 24, 3, 265–273.

[4] Tao, H. and Adams, C. 2008. Pass-Go: A proposal to improve the usability of graphical passwords. *Int. Journal of Network Security*. 7, 2, 273-292.

[5] Zhao, Z., Ahn, G.-J., Seo, J.-J., and Hu, H. 2013. On the security of Picture Gesture Authentication. In *USENIX Security 2013*.

[6] Uellenbeck, S., Dürmuth, M., Wolf, C., and Holz, T. 2013. Quantifying the security of graphical passwords: the case of android unlock patterns. In *ACM CCS'13*. 161-172.

[7] Chiasson, S., Forget, A., Biddle, R., and van Oorschot, P. C. 2008. Influencing users towards better passwords: Persuasive cued click-points. In *Proc. of HCI*. British Computer Society.

[8] Chiasson, S., Stobert, E., Forget, A., Biddle, R., and van Oorschot, P. C. 2012. Persuasive cued click-points: Design, implementation, and evaluation of a knowledge-based authentication mechanism. *IEEE Trans. on Dependable and Secure Computing*. 9, 2 (March/April 2012), 222-235.

[9] Wiedenbeck, S., Waters, J., Birget, J. C., Brodskiy, A., and Memon, N. 2005. Authentication using graphical passwords: Effects of tolerance and image choice. In *Proc. Symp. on Usable Privacy and Security (SOUPS'05)*.

[10] Wiedenbeck, S., Waters, J., Birget, J. C., Brodskiy, A., and Memon, N. 2005. PassPoints: Design and longitudinal evaluation of a graphical password system. *Int. Journal of Human-Computer Studies (Special Issue on HCI Research in Privacy and Security)*. 63, 102–127.

[11] Chiasson, S., van Oorschot, P. C., and Biddle, R. 2007. A second look at the usability of click-based graphical passwords. In *Proc. Symp. on Usable Privacy and Security (SOUPS'07)*.

[12] Golofit, K. 2007. Click passwords under investigation. In *12th European Symposium on Research in Computer Security (ESORICS'07)*. LNCS vol. 4734 (Sept. 2007).

[13] Dirik, A., Memon, N., and Birget, J. 2007. Modeling user choice in the PassPoints graphical password scheme. In *Proc. Symp. on Usable Privacy and Security (SOUPS'07).*

[14] Chiasson, S., Forget, A., Biddle, R., and van Oorschot, P. C. 2009. User interface design affects security: Patterns in click-based graphical passwords. *Int. Journal of Information Security*. Springer, 8, 6 (2009), 387-398.

[15] van Oorschot, P. C., and Thorpe, J. 2011. Exploiting predictability in click-based graphical passwords. *Journal of Computer Security*. 19, 4 (2011), 669-702.

[16] Thorpe, J., and van Oorschot, P. C. 2007. Human-seeded attacks and exploiting hot-spots in graphical passwords. In *USENIX Security*'07.

[17] Salehi-Abari, A., Thorpe, J., and van Oorschot, P. C. 2008. On purely automated attacks and click-based graphical passwords. In *Proc. 24th Annual Computer Security Applications Conference (ACSAC'08)*.

[18] van Oorschot, P. C., Salehi-Abari, A., and Thorpe, J. 2010. Purely automated attacks on PassPoints-style graphical passwords. *IEEE Trans. Information Forensics and Security*. 5, 3 (2010), 393-405.

[19] Chiasson, S., van Oorschot, P. C., and Biddle, R. 2007. Graphical password authentication using cued click points. In *ESORICS'2007*. LNCS, vol. 4734/2007, 359–374.

[20] Juels, A. and Rivest, R. L. 2013. Honeywords: Making password-cracking detectable. In *ACS CCS 2013*.

[21] Jermyn, I., Mayer, A., Monrose, F., Reiter, M., and Rubin. A. 1999. The design and analysis of graphical passwords. In *8th USENIX Security 1999*.

[22] Blonder, G. 1996. *Graphical Passwords*. United States Patent 5559961.

[23] Monrose, F. and Reiter, M. K. 2005. Graphical passwords. *Security and Usability*. L. Cranor and S. Garfinkel, editors. O'Reilly, Chapter 9, 147–164.

[24] Biddle, R., Chiasson, S., and van Oorschot, P. C. 2012. Graphical passwords: Learning from the first twelve years. *ACM Computing Surveys*. 44, 4, Article 19, 1-41.

[25] B. B. Zhu, D. Wei, M. Yang, and J. Yan. 2013. Security implications of password discretization for click-based graphical passwords. In *WWW 2013*.

[26] Khosla, A., Xiao, J., Torralba, A., and Oliva, A. 2012. Memorability of image regions. In *Advances in Neural Information Processing Systems*, 305-313.

[27] Judd, T., Ehinger, K. A., Durand, F., Torralba, A. 2009. Learning to predict where humans look. In *Int. Conf. on Computer Vision (ICCV'09)*. 2106-2113.

[28] Ling, H., and Okada, K. 2006. Diffusion distance for histogram comparison. In *IEEE. Computer Vision and Pattern Recognition (CVPR'06)*. 1, 246-253.

[29] Goldberg, A. V., and Radzik T. 1993. A heuristic improvement of the Bellman-Form algorithm. *Applied Mathematics Letters*. 6, 3, 3-6.

[30] Birget, J. C., Hong, D., and Memon, N. 2006. Graphical passwords based on robust discretization. *IEEE Trans. Information Forensics and Security*. 1, 3 (2006), 395-399.

[31] Chiasson, S., Srinivasan, J., Biddle, R., and van Oorschot, P. C. 2008. Centered discretization with application to graphical passwords. In *Proc. 1st Conf. on Usability, Psychology, and Security (UPSEC'08)*.

[32] Bonneau. J. 2012. The science of guessing: Analyzing an anonymized corpus of 70 million passwords. In *IEEE Symp. on Security and Privacy*.

[33] Mazurek, M.L., Komanduri, S., Vidas, T., Bauer, L., Christin, C. Cranor, L.F., Kelley, P.G., Shay, R., and Ur, B. 2013. Measuring password guessability for an entire university. In *ACM CCS'13*. 173-186.

# 11. APPENDIX

## 11.1 Representative Attack Results

For easy comparison, Table 4 summarizes representative results of all prior attacks on click-based graphical passwords, together with the results of our attacks on PCCP presented in this paper. The first two columns list attack methods and their targeted graphical password schemes, respectively. The next three columns list success rates they achieved, sizes of their attack dictionaries and of their theoretical password spaces, respectively.

**Table 4: Attacks and representative results**

| Attack Method | Scheme | Success rate (%) | Dictionary size (bits) | Theoretical space (bits) |
|---|---|---|---|---|
| Automated [13] | PassPoints | 8.45 | 31.6 | 40 |
| | | 61* | 24.8* | |
| Automated [16] | PassPoints | 0.9 – 9.1 | 35 | 43 |
| Automated [17] | | 8-15 | 24.6 | |
| | | 16 | 31.4 | |
| Automated [18] | | 7-16 | 26 | |
| | | 48-54 | 35 | |
| Human-seeded [15,16] | | 20 to 36 | 31 to 33 | |
| | | 4 to 10 | 6-7 | |
| Our automated | PCCP | 45.83 | 35 | |
| | | 5.21 | 32 | |
| Our human-assisted | | 55.21 | 35 | |
| | | 2.08 | 25 | |

*This result was obtained with image Bird shown in Figure 4, which is much simpler than the images used in other studies.

## 11.2 Distinguishability via Color Dissimilarity

### *11.2.1 Color Dissimilarity of Two Regions*

Dissimilarity of two regions can be compared by their color distributions, i.e., histograms. Ling et al. [28] proposed a metric to measure dissimilarity between two histograms by applying a diffusion process to the difference of two histograms. By modeling this diffusion approach, we define a metric $D_s$ to measure dissimilarity of regions $A$ and $B$ of size $n \times n$ for one color component:

$$D_s(A,B) = \min_f \sum_{x,y} |A(x,y) - B\big(f(x,y)\big)|, \quad (3)$$

where $f$ is a one-to-one mapping function between pixels in regions $A$ and $B$.

The problem to find the optimal $f$ in Eq. (3) can be converted to a minimum-cost maximum-flow problem as follows: a directed graph $G$ comprises $2 + 2n^2$ nodes, i.e., a source $s$, a sink $t$, and one node per pixel in two regions. There is an arrow from source $s$ to each node in $A$ and an arrow from each node in $B$ to sink $t$, each arrow has capacity 1 and cost 0. For each node in $A$, there is an arrow from the node to every node in $B$, with capacity 1 and cost equal to the absolute difference of their pixel values. In addition, for each aforementioned arrow, there is a reverse arrow pointing to the opposite direction with capacity 0 and cost being the opposite (i.e., negative) of the aforementioned arrow's cost.

A method to solve the min-cost max-flow problem is a heuristically improved Bellman-Form algorithm [29]. The set of edges from A's node to B's node without any residual capacity in the optimal solution defines an optimal one-to-one mapping function $f$ for Eq. (3), which is then used to calculate $D_s(A,B)$ with Eq. (3).

We use Lab and Luv, two different color spaces, to calculate region dissimilarity, in hope that dissimilarity manifests in at least one color component. The Euclidean distance of five dissimilarity values each calculated in one of the five color components: L, a, b, u, v. is used as a metric of color dissimilarity of the two regions.

### *11.2.2 M-Index by Color Dissimilarity*

To calculate color dissimilarity of a detectable point $p$, we apply a sliding window $W_s$ of size n × n to move around $p$, with its center being inside or on the boundary of a neighboring region $\mathbb{N}_p$ of size $(2N+1) \times (2N+1)$ centered at $p$, starting at $(-\mathrm{N}, -\mathrm{N})$ relative to $p$ and with a moving step of $\Delta$ pixels along either direction. At each stop, unless $W_s$ is too close to $p$, we calculate the color dissimilarity between $W_s$ and the region $W_p$ of size n × n centered at $p$. More specifically, the color dissimilarity between $W_s$ and $W_p$ is calculated if the center of $W_s$ has a Manhattan distance from $p$ not less than a threshold $d_M$ and an Euclidean distance from $p$ not less than another threshold $d_E$. Regions very close to $W_p$ are excluded since the range examined by dissimilarity should be significantly larger than that used in corner detection wherein immediate neighborhood is used. The average of the calculated dissimilarity values, denoted by $\overline{S_C}$, is used as a metric of color dissimilarity of detectable point $p$.

$\lambda_{color}$ in Eq. (1) is a step-wise factor determined by the color dissimilarity of the point: 1) $\lambda_{color} = \lambda_1 > 1$ to increase M-index

for small dissimilarity: $\overline{S_C} < S_1$ for a preset threshold $S_1$, whereof $p$ is considered unlikely distinguishable; 2) $\lambda_{color} = \lambda_2 < 1$ to lower M-index for large dissimilarity: $\overline{S_C} > S_2$, whereof $p$ is considered likely distinguishable; and 3) $\lambda_{color} = 1$, i.e., no change, for dissimilarity falling between the two cases.

In our experiments to be reported, we used an empirical approach to determine aforementioned parameters: we selected representative points on several typical images, and compared their relative orders given by human labeled results (see Section 4.2) with those by M-indices calculated with Eq. (1) using different values of the parameters. The set of values that produced best match was selected. They were: n = 7, N = 15, $\triangle = 5$, $d_M = 10$, $d_E = 7$, $S_1$=240, $S_2 = 310$, $\lambda_1 = 2$, and $\lambda_2 = \frac{1}{\sqrt{2}} \approx 0.71$.

## 11.3 Distinguishability via Gradient Dissimilarity

We first calculate gradients of an image. For each detectable point $p$, we use a rectangle $R$, with its short edge centered at $p$, to rotate around $p$. Starting at 0°, we calculate the average of gradients in $R$ per every $\delta$ degree, resulting in $m = \lfloor 360/\delta \rfloor$ values at $m$ different angles around $p$.

Uniformity of the $m$ values at different angles indicates how similar the point's neighborhood is. We use a simple empirical metric to measure this uniformity: ratio $r$ of the maximum to the average of the $m$ values. A large $r$ means a large change of gradients at different directions around $p$.

$\lambda_{grad}$ in Eq. (1) is a step-wise factor determined by ratio $r$: 1) $\lambda_{grad} = \beta_1 \gg 1$ to increase M-index significantly for small $r$: $r < r_1$ where $r_1$ is a preset threshold, whereof the point's neighborhood is so uniform that the point is considered very unlikely distinguishable; 2) $\lambda_{grad} = 1$, i.e., no change, for large $r$: $r > r_2$, wherein the point's neighborhood is very dissimilar, and the point is very likely distinguishable, and 3) $\lambda_{grad} = \beta_2 > 1$ to increase M-index slightly for $r$ between the two cases.

In our experiments, the above parameters were determined in the same way as those in the color dissimilarity. The following values were used: the size of rectangle $R$ was $3 \times 15$; $\delta = 8^\circ$, resulting in $m = 45$; $r_1 = 1.45$, $r_2 = 1.8$; $\beta_1 = 10$, and $\beta_2 = 2$.

## 11.4 Adjusting M-Index by Click Patterns

Line is detected by using least-squares to fit 5 points in a word with a straight line. If the square root $e$ of the least square error is less than a preset threshold $e_T$, $e < e_T$, and the word's sequence follows a consistent direction, the word is determined to exhibit Line with a confidence level:

$$c_L = 1 - e/e_T, \quad 0 \le c_L \le 1, \tag{4}$$

and its M-index is multiplied by a factor $\kappa_L \le 1$ for the automated memorability or subtracted by a subtrahend $5\zeta_L \ge 0$ for the human-assisted memorability, where $\zeta_L$ is the subtrahend per point in the word. Both $\kappa_L$ and $\zeta_L$ depend on confidence level $c_L$.

For a sequence of points, one point to the next point forms a directional line. Each pair of adjacent directional lines forms an angle. Regular is detected by checking consistency of angles of adjacent directional lines in a word. Excluding angles very close to 0 degree, if the remaining angels are all positive or negative, then the sequence is counter clock-wise or clock-wise. If they have mixed signs but their absolute values are all smaller than a threshold, the sequence is roughly alone one direction. In both cases, if the standard deviation $\delta$ of all the angles is less than a preset threshold $\delta_T$, the word is determined to exhibit Regular with a confidence level:

$$c_R = 1 - \delta/\delta_T, \quad 0 \le c_R \le 1, \tag{5}$$

and its M-index is multiplied by a factor $\kappa_R \le 1$ for the automated memorability or by subtracting $5\zeta_R \ge 0$ for the human-assisted memorability. Again, both $\kappa_R$ and $\zeta_R$ depend on confidence level $c_R$.

In our experiments, Line was set twice the effect as Regular at the same confidence level. In addition, Regular at confidence level 1 was set to match multiplication factor $\lambda_2$ in Appendix 11.2 for the automated memorability, and Line was set to match the adjusted range in Section 4.3 per point for the human assisted memorability. The parameters were then calculated as: $\kappa_L = 1 - 0.5c_L \in [0.5,1]$ and $\kappa_R = 1 + c_R\left(\frac{1}{\sqrt{2}} - 1\right) \in [\frac{1}{\sqrt{2}}, 1]$; $\zeta_L = c_L \in [0,1]$ and $\zeta_R = 0.5c_R \in [0, 0.5]$.