Every Shred Helps: Assembling Evidence from Orphaned JPEG Fragments

Emre Durmus, Paweł Korus, Nasir Memon, *Fellow, IEEE*

***Abstract*—In this study, we address the problem of forensic photo carving, which serves as one of the key sources of digital evidence in modern law enforcement. We propose efficient algorithms for assembling meaningful photographs from orphaned photo fragments, carved without access to file headers, meta-data or compression settings. The addressed problem raises a novel variant of a jigsaw puzzle with unknown number of mixed images, missing pieces, and severe brightness and colorization artifacts. We construct an efficient compatibility metric for matching puzzle pieces, and a corresponding image stitching procedure which allow to mitigate these artifacts. To facilitate photo assembly, we perform forensic analysis of the fragments to provide clues about their location within the frame of the imaging sensor. The proposed algorithm formulates the assembly problem as finding non-overlapping sets in an interval graph spanned over the input fragments. The algorithm exhibits lower computational complexity compared with a popular puzzle solving approach based on minimal spanning trees.**

I. Introduction

As the number of digital devices in use continues to increase, law enforcement agencies struggle to keep up with the growing demand for forensic analysis of seized units. In cases where the file system information is missing or corrupted, evidence recovery involves a process known as file carving [1, 2]. Carving techniques are also increasingly adopted to volatile memory forensics [3, 4] to recover meaningful pieces of digital objects [5]. Recently researchers have demonstrated impressive tools for extracting widgets from graphical user interfaces in Android applications [6], or for carving uncaptured photographs from volatile memory buffers [7].

This paper focuses on carving digital photographs, which grows in significance due to exponentially increasing number of camera-enabled devices (e.g., smart-phones, smart doorbells, child monitors, smart refrigerators). Such devices perform frequent storage and deletion of captured photos in their flash memory, which increases the likelihood that forensic analysis will run into deleted, incomplete and highly fragmented data. Conventional carving software relies on known signatures of file headers and footers to identify digital images without access to file-system's meta-data. Such techniques often prove unreliable when dealing with fragmented files, especially when pieces are missing. The problem is particularly important for solid state drives (used in mobile devices and increasingly adopted in personal computers) which introduce additional address space randomization to alleviate memory cell wear [8]. The goal of our work is to enable extraction of reliable evidence in such challenging scenarios. In sensitive, high-profile cases like child sexual abuse investigations, every shred of evidence can make a difference.

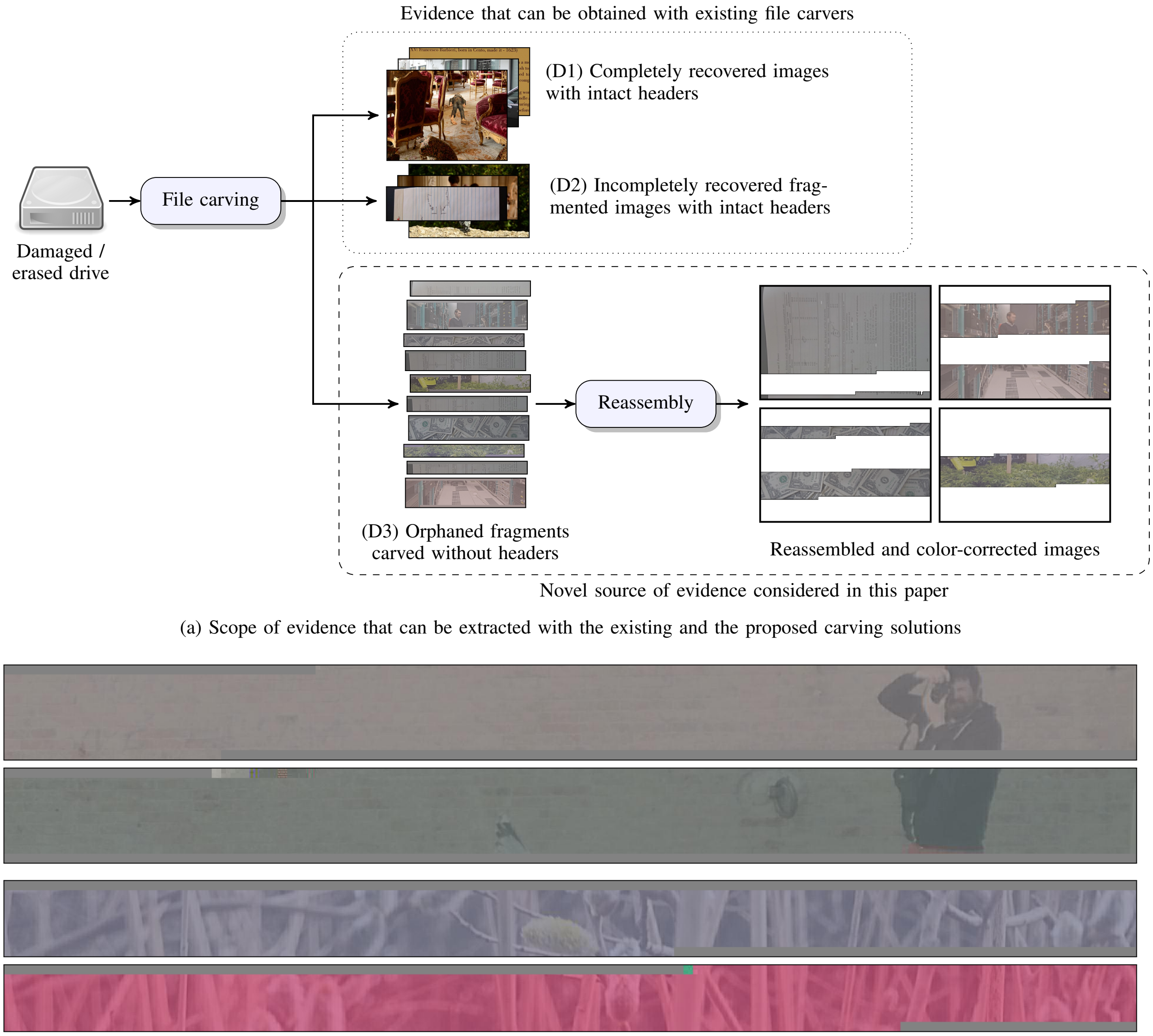
In Fig. 1a we have illustrated existing and the considered sources of photographic evidence that can be obtained via photo carving. We distinguish three levels of difficulty: *(D1)* continuously stored images for which file headers/footers can be identified; *(D2)* fragmented images for which file header (and possibly footer) can be identified and individual fragments are stored at progressively growing addresses; *(D3) orphaned fragments* of images extracted without file headers. The former case is trivial and can be handled by any file carving software. The second case can be handled by state-of-the-art photo carvers and requires sophisticated algorithms for efficient handling of file fragmentation. To the best of our knowledge, only two software packages currently deliver this functionality: (1) the well established *Adroit Photo Forensics* which uses a smart-carving technology based on sequential hypothesis testing [9]; and (2) the recent *JPGCarve* which includes improvements to limit the search space and enhance detection of fragment discontinuities [10]. However, even these tools cannot handle missing fragments (they stop at the first discontinuity).

Orphaned photo fragments (D3) constitute a novel source of evidence that is currently untapped by any photo carving software. Such fragments are carved without access to file headers, which contains important information both about data organization within the bit-stream (e.g., color sub-sampling) and about the raw content itself (e.g., quantization matrices). Without this knowledge, carving was considered infeasible. In a recent paper, Uzun et al. [11] have demonstrated a proof-of-concept prototype for orphaned photo carving. The authors have analyzed a large corpus of image meta-data and collected statistics about the most frequently used compression settings. This information is then used for repeated content reconstruction attempts, followed by sanity checks based on the expected appearance of natural images. The extracted fragments are characterized by colorization and brightness artifacts, cyclic shifts along image width, and occasional damaged blocks at the beginning/end of the data segments (see Fig. 1b).

While extraction of orphaned fragments is an important step forward, it does not by itself represent a reliable and convenient form of evidence. Manual analysis of the fragments and their assembly into comprehensible images is prohibitively time-consuming. While the problem bears resemblance to jigsaw puzzle solvers considered in computer vision, there are important differences which prevent their direct application. The considered jigsaw puzzle is one of the most challenging

E. Durmus and N. Memon are with Tandon School of Engineering, New York University, USA. (e-mail: {edurmus,memon}@nyu.edu).

P. Korus is with Tandon School of Engineering, New York University, USA and also with the Department of Telecommunications, AGH University of Science and Technology, Poland. (e-mail: pkorus@nyu.edu).


Evidence that can be obtained with existing file carvers
(D1) Completely recovered images with intact headers
(D2) Incompletely recovered fragmented images with intact headers
File carving
Damaged / erased drive
Reassembly
(D3) Orphaned fragments carved without headers
Reassembled and color-corrected images
Novel source of evidence considered in this paper

(a) Scope of evidence that can be extracted with the existing and the proposed carving solutions

(b) Example distortions in orphaned photo fragments (two neighboring fragments from two images)

Fig. 1. Illustration of various sources of carved photographic evidence: (top) scope of evidence that can be extracted by existing file carving software and by the proposed approach; (bottom) example distortions in carved orphaned photo fragments; colorization and brightness artifacts stem from missing file header and context information.

variants reported in the literature. The puzzle contains a mixture of fragments from multiple images, whose number is unknown, and whose pieces are potentially missing. Moreover, the fragments suffer from colorization and brightness artifacts.

Our goal is to address these problems, and propose effective algorithms for assembling useful evidence from orphaned photo fragments. To facilitate the assembly process, we exploit additional clues about the fragments that stem from their forensic analysis. Specifically, we consider analysis of photo-response non-uniformity (PRNU) which allows to precisely locate the fragments within the original frame of the imaging sensor. Such analysis requires a precomputed fingerprint of the camera, which is available or can be obtained in several important forensic scenarios and can also be estimated from full images extracted by existing carvers (difficulty levels D1 and D2). This idea builds upon recent preliminary work, where a simple greedy heuristic assembler was considered for a less challenging variant of the problem [12]. The previous study did not allow for flexible control of assembling trade offs and ignored brightness and colorization artifacts. It relied on a histogram-based compatibility function, which does not allow for efficient artifact compensation. Hence, in real-world conditions it will deliver sub-optimal performance.

The proposed algorithm is based on an interval graph formulation of the assembly problem. To the best of our knowledge, this is the first use of such an approach for the problem at hand. While the developed algorithm is still greedy, it does not require global knowledge about the similarity of individual pieces. Compatibility relations between fragments are computed only when needed, which leads to significantly lower computational complexity compared with popular greedy approaches discussed in the literature - even when taking into account problem-specific optimizations.

The main contributions of our work include:

1) proposition of a novel variant of the jigsaw puzzle problem with direct applicability in forensic science;
2) construction of an effective solution for dealing with colorization and brightness artifacts in orphaned photo fragments;
3) proposition of a novel algorithm for image assembly based on interval graphs;
4) adoption of forensic analysis of image fragments to deliver additional information about the puzzle;
5) experimental evaluation of the proposed approach on a large corpus of simulated carved fragments.

The paper is organized as follows. Section II reviews related work and briefly introduces fundamental concepts. Section III formally formulates the problem, and describes the proposed solution. Section IV presents experimental evaluation result. Finally, we conclude and discuss the perspectives for future research in Section VI. The source code for our method will be shared with the research community on request.

II. Background and Related Work

In this section, we review related work in forensic photo carving and assembling jigsaw puzzles. In particular, we focus on the most recent technique for carving orphaned photo fragments. We also briefly introduce the fundamentals of interval graph theory and forensic analysis of sensor pattern noise, which are the building blocks of the considered approach.

A. Forensic Photo Carving

Without data fragmentation, carving digital photographs can be effective with general techniques based on known header/footer signatures. However, fragmentation occurs not only for deleted files, but also naturally due to file-system design. Dealing with fragmented images requires in-depth knowledge of both the storage format's syntax and the expected image statistics. For more information about the challenges and the adopted techniques, the readers are referred to a survey paper [2].

a) Fragmented Photographs: One of the first effective techniques was proposed by Garfinkel [13] and allowed to deal with bi-fragmented JPEG images. After identifying the header and the footer, the method attempts to decode the data in between by excluding a variable size gap in the middle. Successful carving of heavily fragmented photos was enabled by Memon et al. [14] who considered each fragment as a node in a fully connected graph, and formulated the problem as finding k-disjoint shortest paths. The edges were weighted based on fragment similarity. Full connectivity of the graph led to poor computational scalability of the algorithm.

The above limitations were addressed by Pal et al. [9] who proposed to perform sequential hypothesis testing to detect fragmentation points and then search subsequent data blocks for the next fragment. The process continues until completion (footer is reached) or failure (maximal number of steps is exceeded). The technology has been marketed as *smart carving* and is featured in the popular commercial photo carving software - Adroit Photo Forensics. Adroit represents the current state-of-the-art in commercially available solutions.

A recent JPGCarve system [10] improves upon the performance of smart carving. The process starts by automatic detection of file-system cluster size, followed by removal of clusters without correctly detectable photo fragments. Comparison of visual content features allows to match adjacent fragment. While the method allows to deal with data fragmentation, successive pieces of the photo need to occur in the original order. A valid header is still required for each carved photo. The JPEG header defines important properties of the bit-stream syntax (e.g., color sub-sampling) and the content itself (e.g., quantization tables).

b) Orphaned Photo Fragments: Successful carving of individual photo fragments without access to the file header has recently been demonstrated by Uzun et al. [11]. The authors examined a large corpus of JPEG images downloaded from Flickr to identify the most common compression settings. The carver uses this information to guess the compression parameters and attempt fragment decoding. Upon successful decoding the content is also validated against the expected natural image statistics.

While the ability to carve orphaned photo fragments is an important step forward, the fragments themselves do not yet represent useful evidence. They need to be first re-assembled into meaningful images, which is excessively time-consuming for humans and constitutes an open research problem. The process is further complicated by visual artifacts in the recovered fragments. Due to missing information about the image structure and content context (quantization tables, brightness offsets), the fragments exhibit colorization and brightness artifacts, and may be cyclically shifted along image width. Moreover, truncated bit-stream may lead to minor decoding errors at the beginning/end of the fragments (occasionally manifested by a few damaged blocks). Visual impact of these artifacts can be observed in example carved fragments in Fig. 1b.

B. Jigsaw Puzzles

Assembling images from shuffled fragments is a well known problem in computer vision, referred to as the *jigsaw puzzle* problem. Several variants of the puzzle have been considered including assembling irregular, shredded [15, 16] and regular, square fragments [17–19]. The former is known to be easier since irregular shape boundaries limit the space of compatible candidates. Other variations include mixed puzzles with several images [20] (with or without knowledge about the number of images in the mixture), missing pieces [18], or constraints on fragment location or orientation [20].

In general, puzzle solvers involve two main components: (1) a *compatibility function* for measuring match quality between a pair of fragments; (2) an *assembly procedure* which groups and stitches the fragments together. Except for shape constraints, compatibility functions typically rely on pixel intensity differences along fragment boundaries and use various p-norms to measure the distance either in the RGB or Lab color spaces [17]. Some approaches predict continuation of the fragment boundary either using inpainting [21] or row differences [17, 18]. It may also be beneficial to match gradients instead of pixel intensities [20].

The jigsaw puzzle problem is known to be NP complete, so all solvers adopt greedy approximation strategies. The most popular approach involves computation of minimal spanning trees in the constructed fragment graph [15, 16, 20]. The algorithms also differ in their choice of the first placed piece, which may be random or deterministic [18]. The puzzle may also be formulated in terms of dynamic programming [22] which simultaneously finds multiple good matches using a well-known Hungarian optimal assignment procedure. The problem can also be approached as a probabilistic graphical model, where the globally optimal configuration of the pieces can be efficiently approximated using loopy belief propagation [23].

C. Analysis of Sensor Pattern Noise

To facilitate the assembly process, we exploit additional information about the image fragments that can be obtained from their forensic analysis. Specifically, we resort to analysis of sensor pattern noise, which characterizes behavior of individual pixels of imaging sensors used in digital cameras. Such analysis allows us to precisely recover spatial locations of the fragments. As a result, we are able to detect overlaps and order the fragments according to their spatial location, which imposes a regular structure upon the problem which can be exploited to speed-up computations (see Section 8).

Analysis of sensor pattern noise relies on intrinsic imperfections of the sensor which manifest themselves as consistent sensitivity bias of individual pixels. This behavior is unique to each sensor and is stable over time. While this phenomenon is practically imperceptible, it can be detected with statistical methods [24–26]. The problem has been widely studied in digital image forensics literature, and relies on a simplified image acquisition model, where a digital photograph is modeled as:

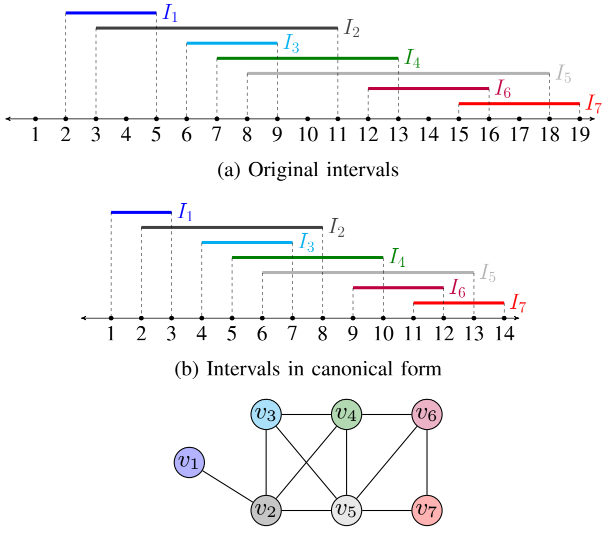
$$I = I^{(0)} + I^{(0)}X + \psi, \tag{1}$$

where $I^{(0)}$ denotes a hypothetical noise-free image, X is the estimated *photo response non-uniformity* (PRNU) noise that represents the camera fingerprint, and ψ combines the remaining sources of noise, e.g., readout noise, dark current, or shot noise. The sensor fingerprint is estimated by averaging noise residuals obtained by subtracting denoised versions of the images.

The fingerprint can be used for attribution and authentication problems, which involve confirmation of the acquisition device or the authenticity of a photo. Verification involves correlation of the investigated image's noise residual with the PRNU fingerprint of the sensor. Brute-force search allows to recover synchronization between the signals which may be necessary for re-sampled or cropped images. The process automatically recovers the original spatial location of the investigated image (or its fragment).

D. Fundamentals of Interval Graphs

Interval graphs have been extensively used for resource allocation problems in operations research and scheduling theory. In this section, we briefly introduce fundamental definitions and concepts to make the paper self-contained.



(a) Original intervals

(b) Intervals in canonical form

(c) Interval graph

Fig. 2. Example intervals and the corresponding interval graph: $\{v_2, v_6\}$ is an example maximal independent set. $\{v_1, v_4, v_7\}$ is an example maximum independent set.

a) Basic Definitions: Let $I = \{I_1, I_2, \ldots, I_n\}$ be closed intervals on the real line $\mathbb{R}$. Then, an *interval graph* G_I corresponding to the interval family I can be defined as $G_I = (V, E)$ where $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{(v_i, v_j) : I_I \cap I_j \neq \emptyset\}$. Two intervals I_j and I_k are *independent* if they do not intersect (see Fig. 2); otherwise they are considered *adjacent* to each other.

A set of intervals S is an *independent set* if all intervals in S are independent of each other. If there is no proper superset of S which is also an independent set, then S is called a *maximal independent set*. If the cardinality of a maximal independent set S is the largest among all independent sets, then it is called a *maximum independent set* (see Fig. 2).

A set of intervals $\{I_1, I_2, ..., I_n\}$ is in *canonical form* if the beginning and ending locations of the intervals are distinct integers between 1 and $2n$ and no two intervals have endpoints at the same locations. The general and the canonical forms share the same set of dependencies and as a result correspond to the same interval graph (see Fig. 2).

b) Finding Maximal Independent Sets: Masuda et al. [27] proposed an algorithm for enumerating all maximal independent sets of an interval graph. The method operates on canonical-form intervals and then constructs a directed graphs whose depth-first-like traversal corresponds to enumeration of the maximal independent sets. Conversion of intervals into their canonical form involves sorting the endpoints and remapping them to integers while preserving overlap relations.

The algorithm scans successive coordinates $i = 1, \ldots, 2n$ and progressively constructs a graph. Let $G_i = (V_i, E_i)$ denote temporary graphs for successive coordinates $i = 1, 2, \ldots, 2n$. The vertex sets V_i consists of a dummy parent node v_0 and nodes v_j corresponding to intervals I_j with starting locations $\leq i$. The algorithms tracks which intervals are immediate predecessors of the current coordinate i, and keeps the corresponding vertices in a *precede set* $\mathcal{P}_i$, e.g., $\mathcal{P}_9 = \{v_2, v_3\}$ in Fig. 2b (v_1 is not included since it is not an immediate

Algorithm 1 Pseudo-code for directed graph creation.

```
Input: $\{I_i\}$                                        ▷ canonical-form intervals
  $V_0 \leftarrow \{v_0\}$, $E_0 \leftarrow \{\emptyset\}$, $G_0 \leftarrow (V_0, E_0)$     ▷ initialize graphs
  $\mathcal{P}_1 \leftarrow \{v_0\}$                                 ▷ intialize precede set
  for $i \leftarrow 1, \ldots, 2n$ do
    $I_j \leftarrow$ interval starting/ending at $i$
    if $i$ is the start of $I_j$ then
      Create a new vertex $v_j$
      $V_i \leftarrow V_{i-1} \cup \{w_j\}$                          ▷ add the vertex
      $E_i \leftarrow E_{i-1} \cup \{(x \rightarrow w_j) | x \in \mathcal{P}_i\}$       ▷ add edges
      $G_i \leftarrow (V_i, E_i)$
      $\mathcal{P}_{i+1} \leftarrow \mathcal{P}_i$
    end if
    if $i$ is the end of $I_j$ then
      $G_i \leftarrow G_{i-1}$
      $l \leftarrow$ start coordinate for $I_j$
      $\mathcal{P}_{i+1} \leftarrow (\mathcal{P}_i - \mathcal{P}_l) \cup \{w_j\}$          ▷ add vertex and prune
    end if
  end for
  return $G_{2n}$
```

predecessor, due to the I_3 in between).

Each processing step depends on whether the current location corresponds to a start or an end of an interval. Upon interval start, a new vertex v_j is created (corresponding to interval I_j), and directed edges are added from each vertex in $\mathcal{P}_i$ to v_j. Upon interval end, v_j is added to the precede set $\mathcal{P}_i$, and the precede set is pruned of nodes whose intervals' ending locations are before the starting location of I_j. Finally, maximal independent sets are enumerated by depth-first-like traversal from v_0 to the vertices in $\mathcal{P}_{2n+1}$ in the last graph G_{2n}.

III. Fragment Assembling Methods

This section introduces the proposed orphaned fragment assembling method based on an interval graph formulation. We also describe a baseline method inspired by existing spanning-tree-based jigsaw puzzle solvers. In either case, the considered fragment assemblers operate in three main steps:

1) Finding vertical positions for all of the fragments based on forensic analysis of sensor pattern noise;
2) Clustering the fragments into separate images based on pairwise fragment compatibility;
3) Stitching the fragments within each cluster along with joint color and brightness normalization.

The interval-graph and spanning-tree assemblers share the same fragment compatibility measure and differ only in the adopted clustering procedure. Both methods incorporate a *compatibility threshold* τ which allows to control the desired matching precision and to trade-off result quality for the number of clusters.

Finding the best clustering of the input fragments can be formulated as an optimization problem which involves placing edges between nodes in a graph spanned over the fragments. The edges should be placed to maximize fragment compatibility:

$$\underset{E}{\operatorname{argmax}} \sum_{(i,j) \in E} \mathcal{C}(F_i, F_j) \tag{2a}$$

$$\text{s.t. } \forall_{i,j} \; \mathcal{C}(F_i, F_j) > \tau \tag{2b}$$

$$\forall_i \; |\{(x,y) \in E : x = i\}| \leq 1 \tag{2c}$$

$$\forall_i \; |\{(x,y) \in E : y = i\}| \leq 1 \tag{2d}$$

where $\mathcal{C}$ denotes the compatibility function assuming values in $\mathbb{R} \cup \{-\infty\}$ ($-\infty$ is used for forbidden connections, e.g., among overlapping fragments). The last two conditions correspond to limits on the in/out degrees of the nodes, and ensure that each vertex can have at most two neighbors (at the top and at the bottom, respectively). The final labeling of the nodes can be obtained by enumerating connected components in the graph.

A. Input and Pre-precessing

The input to the assembler is an unordered and mixed set of orphaned photo fragments $\{F_i\}$ (see Fig. 1b), carved from damaged JPEG photographs using the tool developed by Uzun and Sencar [11]. The fragments are always of the same width (full width of the original image) but vary in height (depending on the size of recoverable data chunks). The fragments can also significantly differ in color and brightness - even in case of direct neighbors. Minor artifacts can also be expected at the beginning/end of the data blocks.

The number of images in the mixture is unknown. Although in general the number of cameras in the mixture is also unknown, we assume that only one camera is involved. To enforce this assumption, the input fragments can be divided into separate groups based on their width and the camera fingerprint match (see Section II-C). In addition to the fragments, our algorithm also requires the PRNU fingerprint of the camera. A high-quality fingerprint can be easily estimated from new photographs if the camera is available. Alternatively, it can also be estimated from existing photos or even from full-size photographs carved by any of the existing tools. Several efficient algorithms exist for blind clustering of large photo collections and joint estimation of their cameras' fingerprints [28].

The main goal of the initial forensic analysis is to determine vertical locations of the fragments. This is performed by brute-force search over all possible locations and correlation of the PRNU fingerprint with noise residuals of the investigated fragments. The search needs to be performed only along the vertical dimension, allowing for rapid computation using standard convolution routines. Comparison of the peak correlation with a threshold allows to detect incorrectly decoded fragments and either discard or correct them (e.g., by reversing cyclic shift along the image width). In our experiments, we used the peak to correlation energy (PCE) metric, known for better stability of the correlation measurements [29].

B. Interval Graph Assembler

The proposed assembler treats each image fragment as an interval with beginning/ending locations recovered as a result of sensor pattern noise analysis (see Fig. 3). To solve

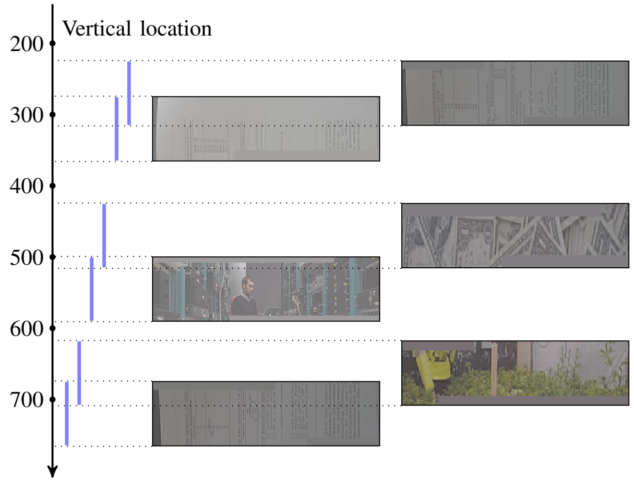


Fig. 3. Carved photo fragments as intervals. Beginning and ending locations are obtained from forensic analysis of sensor pattern noise.

the optimization problem 2, we have implemented a greedy algorithm inspired by Masuda's procedure for enumerating independent sets in an interval graph (Section II-D). Each identified independent set corresponds to an image, which guarantees that overlapping fragments will not be clustered together and allows to quickly find a good approximation of the optimal clustering.

At first, the input fragments are sorted by vertical location and converted into canonical-form intervals. Then, the algorithm scans successive coordinates $i = 1, \dots, 2n$ and constructs a directed graph by matching compatible fragments. Upon start of a new interval, a new vertex v_j is added to the graph. Initially, the algorithm tries to match the new fragment to its immediate predecessors (tracked within precede sets $\mathcal{P}_i$). For each candidate vertex in the precede set $\mathcal{P}_i$, the algorithm computes the compatibility function and the best match is chosen. The matching process may involve breaking existing parent-child relations if the new vertex would constitute a better match. In such a case, the child is detached and moved to a compatible predecessor (from its corresponding precede set) or used to start a new cluster.

If none of the immediate predecessors of v_j constitutes a good match, the algorithm starts checking earlier ancestors. To limit computational complexity, the first acceptable match is used and only a fraction of available candidates is checked. Again, if the match would improve upon existing parent-child relations, the child is detached and used to start a new cluster. Finally, this procedure yields a graph where independent sets form unconnected chains. The flowchart of the algorithm is shown in Fig 4.

C. Spanning Tree Assembler

As a baseline algorithm we consider a minimal spanning-tree approach which was commonly used in earlier jigsaw puzzle solvers [15, 16, 20]. Our assembler adopts the Kruskal's algorithm [30] which performs greedy fragment stitching with the best available pair (above a certain affinity threshold τ). The algorithm begins with each fragment in a separate tree, and then progressively merges the trees (overlapping candidate fragments are simply skipped). In the process, labels are swapped for the smaller available one, which guarantees that the output labels simply enumerate the identified images. The algorithm is summarized as Algorithm 2.

Algorithm 2 The spanning tree assembler.

```
Input: $\{F_i\}$                                   ▷ unordered orphaned fragments
Input: $\tau$                                      ▷ threshold
Input: LUT                                         ▷ lookup table
  $\mathbf{C} \leftarrow \mathcal{C}(\{F_i, F_j\}, \mathrm{LUT})$      ▷ compute compatibility matrix
  $\mathcal{I} \leftarrow \mathrm{sort}(C)$        ▷ order fragment pairs by compatibility (descending)
  $\mathcal{L} \leftarrow [1, \dots, N]$           ▷ initialize labels (fragments in separate cluster)
  $k \leftarrow 0$                                 ▷ pointer to the current pair
  while true do
    $k \leftarrow k + 1$
    $(i, j) \leftarrow \mathcal{I}_k$              ▷ fetch the current pair
    if $A_{ij} \leq \tau$ then
      break                                        ▷ stop if no better affinities available
    end if
    $\hat{\mathcal{L}} \leftarrow \mathcal{L}$     ▷ remember the current labeling
    $l_t \leftarrow \min(\mathcal{L}_i, \mathcal{L}_j)$   ▷ target label for the current pair
    $l_s \leftarrow \max(\mathcal{L}_i, \mathcal{L}_j)$   ▷ source label for the current pair
    if $l_s = l_t$ then
      continue                                     ▷ already processed, skip to the next pair
    end if
    $\{\mathcal{L}_n : \mathcal{L}_n = l_s\} \leftarrow l_t$   ▷ replace the source with target labels
    $\mathcal{M} \leftarrow \{n : \mathcal{L}_n = l_t\}$      ▷ find all fragments with the current label
    if $|\{(x, y) \in \mathcal{M} \times \mathcal{M} : C_{xy} = -\infty\}| > 0$ then
      $\mathcal{L} \leftarrow \hat{\mathcal{L}}$   ▷ found overlap, restore previous labels
      continue                                     ▷ skip to the next pair
    end if
  end while
  return $\mathcal{L}$
```

D. Fragment Affinity Measures

Designing an efficient compatibility measure is a key problem in image re-assembly. As already discussed in Section II-B, the most popular and already very effective approach involves a simple L_p norm of pixel differences along the boundaries between the fragments (computed either in the RGB or the Lab color space) [17]. Such an approach is not directly applicable due to the artifacts in the carved fragments (see Section II-A and Fig. 1). Missing pieces also make it impractical to consider more advanced measures like gradient compatibility [20] or differences of predicted content continuation beyond the fragment boundary [17]. Hence, a new metric is needed.

Computation of fragment compatibility relations is the most time-consuming operation in the whole assembly process. As a result, to minimize computational complexity, we resort to a compensated variant of the L_2 norm. Specifically, we compute the compatibility between fragments F_i and F_j as follows:

$$\mathcal{C}(F_i, F_j) = C_{ij} = \begin{cases} -\infty & \text{if } F_i \text{ and } F_j \text{ overlap,} \\ \log\left(\frac{\hat{C}_{ij}}{1 - \hat{C}_{ij}}\right) & \text{otherwise.} \end{cases}$$

where $\hat{C}_{ij} \in [0, 1]$ represents the probability that the fragments originate from the same image. We estimate this probability from pixel differences D_{ij} (formally defined later in the text) given the spatial distance S_{ij} between the two fragments. For each spatial distance S_{ij}, we obtained empirical probability distributions $p(d|H_0, S_{ij})$ and $p(d|H_1)$ which correspond to

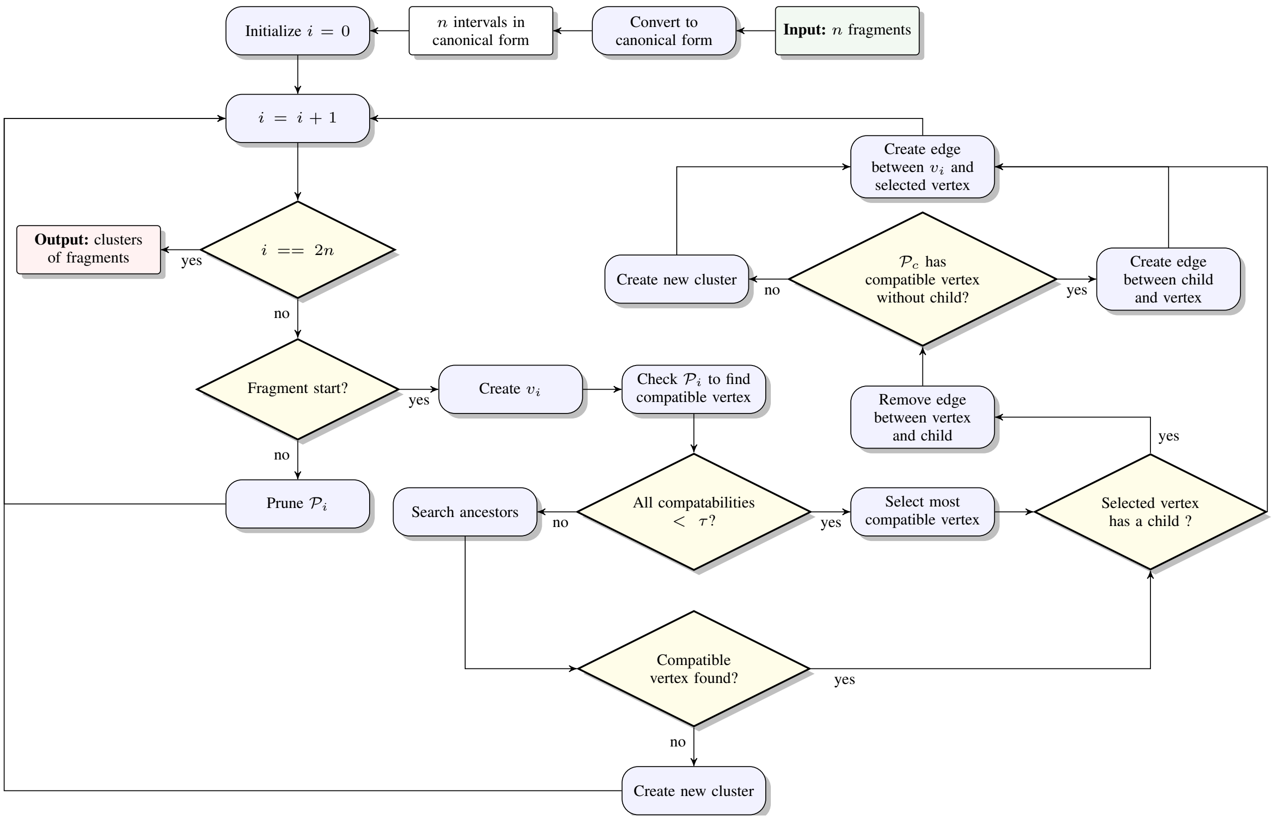

Initialize $i = 0$
n intervals in canonical form
Convert to canonical form
Input: n fragments
$i = i + 1$
Create edge between v_i and selected vertex
Output: clusters of fragments
yes
$i == 2n$
Create new cluster
no
$\mathcal{P}_c$ has compatible vertex without child?
yes
Create edge between child and vertex
no
no
Fragment start?
yes
Create v_i
Check $\mathcal{P}_i$ to find compatible vertex
Remove edge between vertex and child
yes
no
Prune $\mathcal{P}_i$
Search ancestors
no
All compatabilities $< \tau$?
yes
Select most compatible vertex
Selected vertex has a child ?
Compatible vertex found?
yes
no
Create new cluster

Fig. 4. Flowchart of the proposed interval graph assembler.

fragments obtained from the same image (hypothesis H_0) or from different images (hypothesis H_1). The obtained distributions for selected distances $S_{ij} \in \{1, 64, 128, 512\}$ [px] are shown in Fig. 5. Then, given the corresponding cumulative distributions $P(d|H_0, S_{ij})$ and $P(d|H_1)$, we compute:

$$\hat{C}_{ij} = \frac{P(D_{ij}|H_0, S_{i,j})}{P(D_{ij}|H_0, S_{i,j}) + P(D_{ij}|H_1)} \tag{3}$$

which represents preference for treating fragments F_i and F_j as originating from the same image. Exploitation of prior knowledge on typical pixel distributions allows to automatically discount false positive matches for distant fragments, which occasionally could take precedence between closer fragments. For the sake of computational efficiency, we implemented the above relationship as a lookup table $(D_{ij}, S_{ij}) \rightarrow \hat{C}_{ij}$. In our study, we considered 256 possible values for D_{ij} and 128 values for S_{ij} (distance was measured in 8 px increments). We obtained the lookup tables from original image fragments. In order to compensate for the loss of contrast in carved fragments, the pixel distance can be multiplied by a constant factor (in our experiments, we used a multiplier of 4 based on empirical comparison of pixel statistics).

Pixel differences D_{ij} are computed from the corresponding top/bottom rows of the fragments, depending on their spatial arrangement. We assume all fragments originate from images of the same size $(N_h, N_w, 3)$. Let $\mathbf{I}^{(i,b)}$ and $\mathbf{I}^{(j,t)}$ be arrays of size $N_w \times 3$ denoting the bottom and top margins of the top and bottom fragments, F_i and F_j respectively (in case of reverse spatial ordering the top/bottom row selection needs to be reversed as well). For the sake of notation simplicity, we will refer to them in short as $\mathbf{I}^{(b)}$ and $\mathbf{I}^{(t)}$. Then, without color compensation, we could compute D_{ij} as:

$$D_{ij} = \|\mathbf{I}^{(b)} - \mathbf{I}^{(t)}\|_2 = \left(\frac{1}{3N_w} \sum_{w=1}^{N_w} \sum_{c=1}^{3} (I_{wc}^{(b)} - I_{wc}^{(t)})^2 \right)^{\frac{1}{2}}$$

For better stability across various image sizes, the summation is normalized by $3N_w$. Such a metric efficiently identifies fragments originating from the same images assuming that no differences in color and brightness are observed (Fig. 6a). However, in orphaned fragments even directly adjacent fragments can significantly differ both in brightness and in color (Fig. 1) which significantly deteriorates detection performance (Fig. 6b).

Color and brightness correction for a pair of fragments F_i and F_j can be implemented by finding offsets $\alpha_i, \alpha_j \in \mathbb{R}^3$, where each component α_{ic} corresponds to a channel of the image (in principle Y, Cb, Cr in JPEG representation, although the correction works equally well in the RGB color space):

$$\underset{\alpha_i, \alpha_j}{\operatorname{argmin}} \sum_{w=1}^{N_w} \sum_{c=1}^{3} \left((I_{wc}^{(b)} + \alpha_{ic}) - (I_{wc}^{(t)} + \alpha_{jc}) \right)^2 . \tag{4}$$

While for actual color and brightness correction it remains important to find both α_i and α_j (see Section III-E for details), for computation of minimum D_{ij} it suffices to consider a

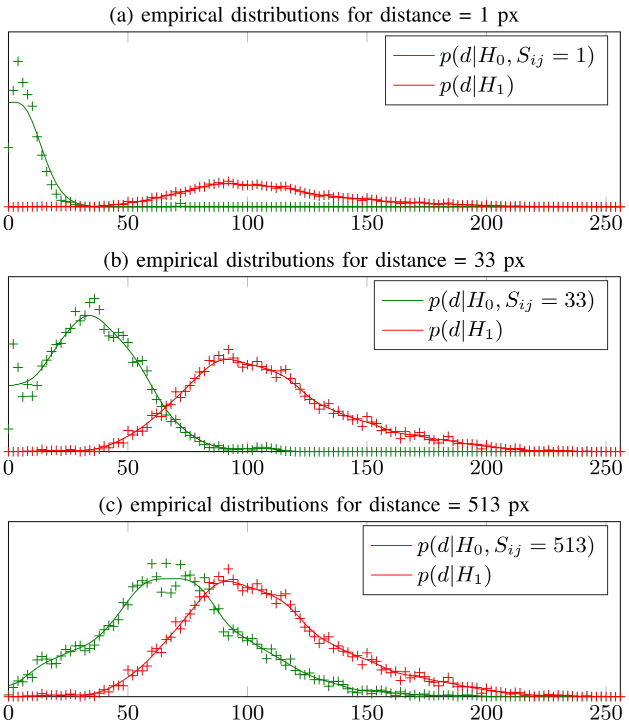


Fig. 5. Empirical distributions of L_2 distances for different-image and same-image fragments separated by three example distances (1, 33, and 513 px).

single offset $\alpha \in \mathbb{R}^3$:

$$\underset{\alpha}{\operatorname{argmin}} \sum_{w=1}^{N_w} \sum_{c=1}^{3} \left(I_{wc}^{(t)} - I_{wc}^{(b)} + \alpha_c \right)^2 \ . \tag{5}$$

A solution to this equation can be obtained by equating its gradient to 0, which yields:

$$\alpha_c = -\frac{1}{N_w} \sum_{w}^{N_w} I_{wc}^{(t)} - I_{wc}^{(b)} \ . \tag{6}$$

The above offsets can then be used to compute $\min_\alpha D_{ij}$ and the corresponding best compatibility $\hat{C}_{ij}$.

Dealing with carved image fragments requires handling missing DCT blocks at the beginning and end of the fragments (see Fig. 1). While exact locations where the data starts/ends are determined by the fragment carver, we assume such information is not available to the re-assembly module and estimate it directly from pixel values. Then, the margins are extracted along the last known valid pixels (both top and bottom margins for each fragment). The shape of the resulting puzzle cannot be used for an exact match, because additional data may be missing or its header/tail may be damaged. However, we consider this shape information to eliminate obvious overlaps between neighboring fragments.

E. Color and Brightness Correction

Color and brightness correction of multiple orphaned fragments relies on similar principles as discussed in Section III-D. However, it is necessary to consider separate offsets for each of the fragments, and the optimization problem (4) needs to be extended to include more constraints. Let us denote fragments assigned to a single image, and ordered according to their vertical location as $F_i : i \in \mathcal{L} = \{1, \ldots, L\}$ and the corresponding correction factors as $\alpha_i \in \mathbb{R}^3$. Then, the problem becomes:

$$\mathcal{O}_{iwc} = \left((I_{wc}^{(i,b)} + \alpha_{ic}) - (I_{wc}^{(i+1,t)} + \alpha_{(i+1)c}) \right)^2 \ , \tag{7a}$$

$$\hat{\alpha}_i = \underset{\alpha_i \ : \ i \in \mathcal{L}}{\operatorname{argmin}} \sum_{i=1}^{L-1} \sum_{w=1}^{N_w} \sum_{c=1}^{3} \mathcal{O}_{iwc} \ . \tag{7b}$$

We solve this system by equating its gradient to 0, which yields the following equations, which combine the constraints from 2 neighborhood relations:

$$-\alpha_{(i-1)c} + 2\alpha_{ic} - \alpha_{(i+1)c} = \left(I_{wc}^{(i-1,b)} - I_{wc}^{(i,t)} \right) - \tag{8a}$$

$$- \left(I_{wc}^{(i,b)} - I_{wc}^{(i+1,t)} \right) \ . \tag{8b}$$

The problem can be conveniently represented by the following linear system (for presentation clarity, we show the solution for a single color channel):

$$\mathbf{M}\alpha = \mathbf{v} \ , \tag{9}$$

where the sought offsets are organized into a column vector:

$$\alpha = \begin{pmatrix} \alpha_{1c} & \alpha_{2c} & \ldots & \alpha_{(L-1)c} & \alpha_{Lc} \end{pmatrix}^T \ , \tag{10}$$

and the matrix $\mathbf{M}$ considers all constraints between neighboring offsets:

$$\mathbf{M} = \begin{pmatrix} 1 & -1 & 0 & 0 & & 0 & 0 \\ -1 & 2 & -1 & 0 & & 0 & 0 \\ 0 & -1 & 2 & -1 & \ldots & 0 & 0 \\ & & \ldots & & & & \\ 0 & 0 & 0 & 0 & & -1 & 1 \end{pmatrix} \ .$$

The free terms are analogously obtained by summing up pixel rows from neighboring fragments with alternating signs:

$$\mathbf{v} = \frac{1}{N_w} \sum_{w} \begin{pmatrix} -\left(I_{wc}^{(1,b)} - I_{wc}^{(2,t)} \right) \\ \left(I_{wc}^{(1,b)} - I_{wc}^{(2,t)} \right) - \left(I_{wc}^{(2,b)} - I_{wc}^{(3,t)} \right) \\ \ldots \\ \left(I_{wc}^{(L-1,b)} - I_{wc}^{(L,t)} \right) \end{pmatrix} \ . \tag{11}$$

The above problem does not have a unique solution ($\det(\mathbf{M}) = 0$) so we obtain the offsets using its pseudoinverse, i.e., $\hat{\alpha} = \mathbf{M}^{+}\mathbf{v}$. Solving for all color channels can be performed jointly by extending the above vectors and repeating matrix $\mathbf{M}$ along the diagonal of a $3L \times 3L$ matrix. An example result of color and brightness normalization with the above algorithm is shown in Fig. 7 for both complete and incomplete images.

F. Computational Complexity

The considered assembly process involves five general steps: (1) localization of the fragments on image canvas; (2) fragment preparation and sorting; (3) affinity computations; (4) clustering the fragments into separate images; (5) stitching and

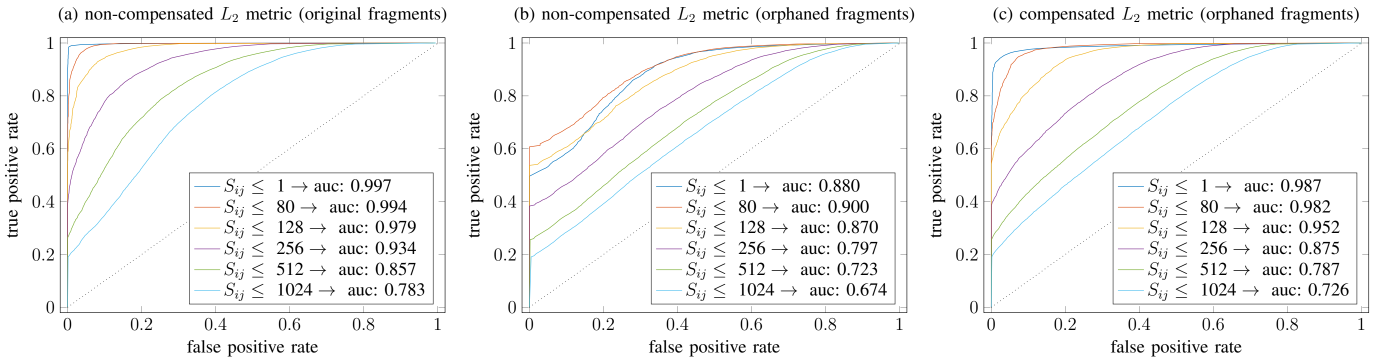


Fig. 6. Receiver operation characteristics for detecting fragments originating from the same images and their changes with spatial distance for: (a) an uncompensated L_2 metric on original fragments; (b) uncompensated L_2 metric on carved orphaned fragments; (c) compensated L_2 metric on carved orphaned fragments.

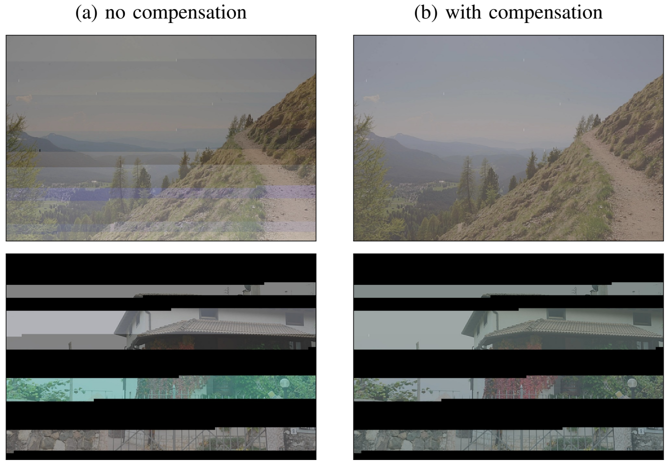


Fig. 7. Result of the considered color and brightness normalization procedure for images re-assembled from orphaned fragments: (top) without missing fragments; (bottom) with missing fragments.

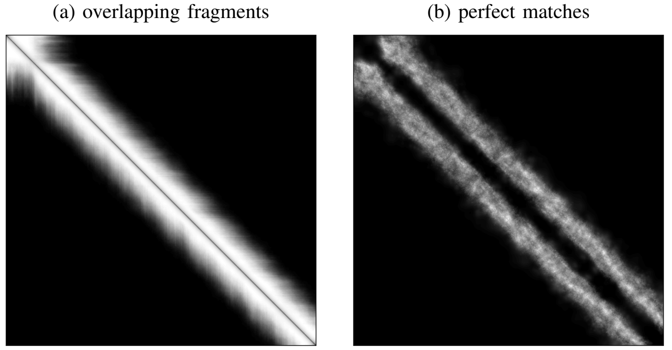


Fig. 8. Structure of the compatibility matrix showing the probability of each (i, j) location being: (a) a pair of overlapping fragments; (b) a perfect match.

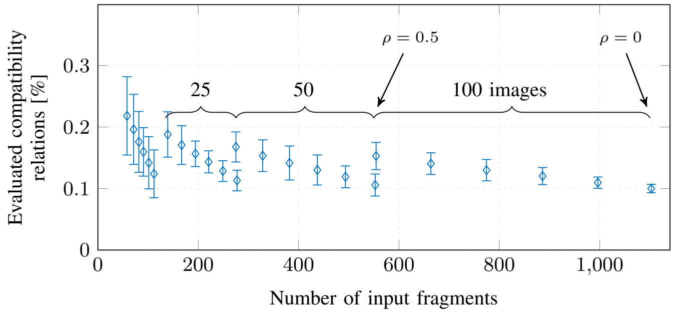


Fig. 9. Fraction of compatibility relations computed by the interval graph assembler.

color and brightness normalization. The most time consuming operations are (1) and (3) which have linear and quadratic complexity, respectively. As a result, computation of fragment affinities is the main bottleneck.

The spanning tree assembler is a greedy algorithm which requires knowledge of all the affinities beforehand ($(n^2 - n)/2$ entries). However, availability of fragment localization information imposes an interesting structure upon the affinity between the fragments. Fig. 8 shows the most probable locations of two important interactions between the fragments: (a) overlapping fragments; (b) fragments with a perfect match according to the affinity metric A_{ij}. The figure was obtained by averaging affinity matrices for various numbers of involved images, various rates of missing fragments, and various repetitions of random phenomena. It can be observed that both of these cases occur in a narrow band around the main diagonal of the matrix. As a result, it becomes possible to severely narrow down the range of indexes where the affinities need to be computed. We observed that limiting the computation to approx. 36 - 43% of all entries (20 - 25% of the edge of the matrix) is enough and does not have negative impact on assembly performance.

The interval graph assembler represents a different approach. While the algorithm is still greedy, it begins with the knowledge of the first top fragment and then computes the affinities only when needed. At first, the algorithm considers only a small set of immediate predecessors, and continues to consider further nodes only if an acceptable match cannot be found. While such an approach significantly limits the amount of necessary computations, the scalability of the algorithm cannot be captured with a simple dependency; important factors include the number of input fragments, the number of images, the number of missing fragments, the decision threshold, and distribution of compatibility values and spatial locations of the fragments.

Fig. 9 shows the typical percentage of all compatibility relations that were computed by the algorithm. We used mixtures of 10, 25, 50 and 100 images and controlled the number of input fragments by randomly removing a given portion of them (from 0 to 50%). The variability stems from various decision thresholds and random repetitions of the experiment. It can be observed that the number of resolved compatibilities grows together with the number of dropped fragments, which corresponds to the lack of good matches among immediate predecessors and a necessity to consider more fragments (hence the discontinuities between cases with the same number of input fragments and different fragment dropping rates). It can also be observed that the percentage of computed compatibilities typically remains between 10-20%, which represents a considerable improvement over the spanning tree approach.

IV. Experimental Evaluation

This section describes the performed empirical validation. We begin by introducing the utilized dataset and explaining the adopted performance metrics. We then discuss the fragment localization performance. Finally, we describe the conducted experiments and discuss the obtained results.

A. Dataset Preparation

We used images from the RAISE dataset which contains 8,000 diverse uncompressed bitmaps acquired by 3 digital cameras [31]. We randomly selected a subset of 2,000 images from one camera and converted them to JPEG with a popular open-source library IJG. We used a single quality level (80) and 4:2:0 chroma sub-sampling. Prior to selection, we did not prune duplicated images (the dataset contains both exact and near-duplicates). These images were then used for: (1) generating simulated fragments for jigsaw reassembly (200 random images); (2) estimating the PRNU fingerprint of the camera (200 most favorable images[1]) using the MLE estimator available at [32]; (3) collecting pixel difference statistics along rows separated by various distances (2,000 images); (4) carving actual orphaned fragments with a prototype carver obtained from the authors of [11] (50 random images).

Most of our experiments were performed on simulated carved fragments which allowed us to operate on a much larger dataset. The simulated fragments replicate the same colorization and brightness artifacts that occur as a result of orphaned photo carving: (1) the first DC coefficients within all color channels were set to zero; (2) DCT coefficients were used directly without dequantization since quantization tables are not available in a general case; (3) first and last rows were zeroed at random truncation locations. To obtain a performance baseline, we also generated corresponding versions of the fragments without color and brightness artifacts (data truncation from step 3 was still performed).

To reduce algorithm runtime and storage requirements, we evaluated assembly performance on down-sampled versions of the images (down to 2 Mpx). Evaluation of forensic fragment analysis and location recovery performance were performed on actual carved fragments obtained from full resolution images.

B. Performance Measures

For each identified cluster of fragments, we determine the identified image as the mode of ground truth image identifiers of the included fragments (the most frequent ground truth identifier is chosen). To measure assembly performance, we then discard the conflicting fragments and compute: (1) the rate of the remaining fragments r_F; (2) the rate of images from the mixture for which at least one cluster has been found r_I; (3) ratio of the number of identified clusters to the number of images in the mixture r_C. We capture the behavior of the assemblers by considering two dependencies: $r_F = f(r_C)$ and $r_I = f(r_C)$ which are empirically obtained by varying the compatibility threshold τ. Such an approach reflects the efficiency of the assemblers in trading off the number of clusters for their purity. In both cases, ideal performance corresponds to the point (1,1).

C. Carving and Pre-processing of Orphaned JPEG Fragments

Carving orphaned photo fragments involves guessing several important compression parameters and hence is susceptible to errors. In this experiment, we evaluate the utility for reassembly of fragments yielded by the orphaned photo carver [11]. We used a research prototype shared with us by the authors. The carver extracts photographic content of a single fragment from a chunk of binary data.

We focus on the ability of precise fragment localization within the sensor frame, which is crucial for our algorithm and reflects the quality of the recovered content. We used 50 full-resolution JPEG images, subsequently divided into fragments by pseudo-random bit-stream truncation. We then ran the orphaned fragment carver to extract photographic content from the truncated bit-streams. The total number of fragments was 1,400 and the number of fragments per image varied between 14 and 48. 1.6% of the fragments were recovered with incorrect image width, and were discarded from further consideration.

We performed fragment localization by 1-dimensional brute-force search based on the PRNU signature (for fragment localization) and on pixel intensities (to obtain ground truth for evaluation). Both search procedures were implemented as cross-correlation using standard fast 2-D convolution routines. For the sensor fingerprint, we used the PCE measure to assess the quality of the match. We used 3 different camera fingerprints, obtained from both uncompressed images, as well as compressed JPEG images with quality factors 90 and 80.

We measured distances between spatial locations recovered based on the sensor fingerprint and the pixels. Since JPEG images are encoded based on macro-blocks (four 8 × 8 px blocks), minor localization errors can be easily compensated. Hence, we consider vertical localization distance below 8 px as successful. We used PCE threshold of 7, to accept carved

[1] We used a standard correlation predictor to estimate the PRNU strength in an image [24]. As expected, the chosen images correspond to bright, but non-saturated, images with little texture.

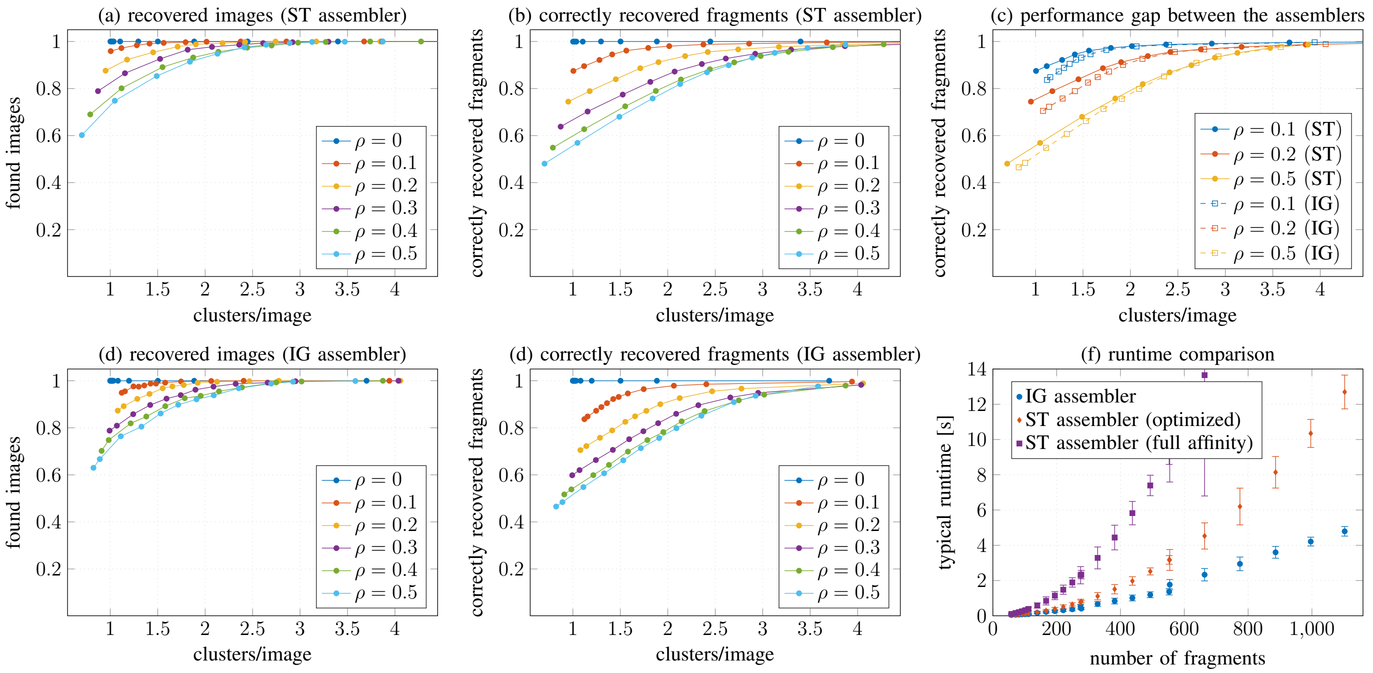


Fig. 10. Fragment assembly performance for the IG and ST assemblers: (left) recovered images r_I; (middle) correctly recovered fragments r_F; (top right) comparison of r_F for selected drop rates; (bottom right) comparison of typical algorithm runtime.

TABLE I
FRAGMENT LOCALIZATION ACCURACY FOR ORPHANED PHOTO FRAGMENTS [%].

	TIFF	JPEG 90	JPEG 80
True positives	56.7	52.4	42.0
False positives	0.8	0.5	0.6
True negatives	31.3	35.8	45.7
False negatives	9.2	9.7	9.6

fragments. The localization success results are collected in Table I.

Assuming a good camera fingerprint is available (uncompressed and JPEG 90 fingerprints in our evaluation), $> 50\%$ of the fragments were localized perfectly. The number of false positives is negligible. Upon manual inspection, all cases (0.6%) turned out to be empty or nearly flat fragments where pixel-based localization failed leading to corrupted ground truth data. In all cases, PRNU-based localization returned correct locations.

Manual inspection of the true negative cases revealed circularly shifted content which prohibited a positive match. While this distortion could be corrected by performing sensor pattern noise validation for plausible cyclic shifts (essentially adding an extra dimension to the search-space), we leave this improvement for future work.

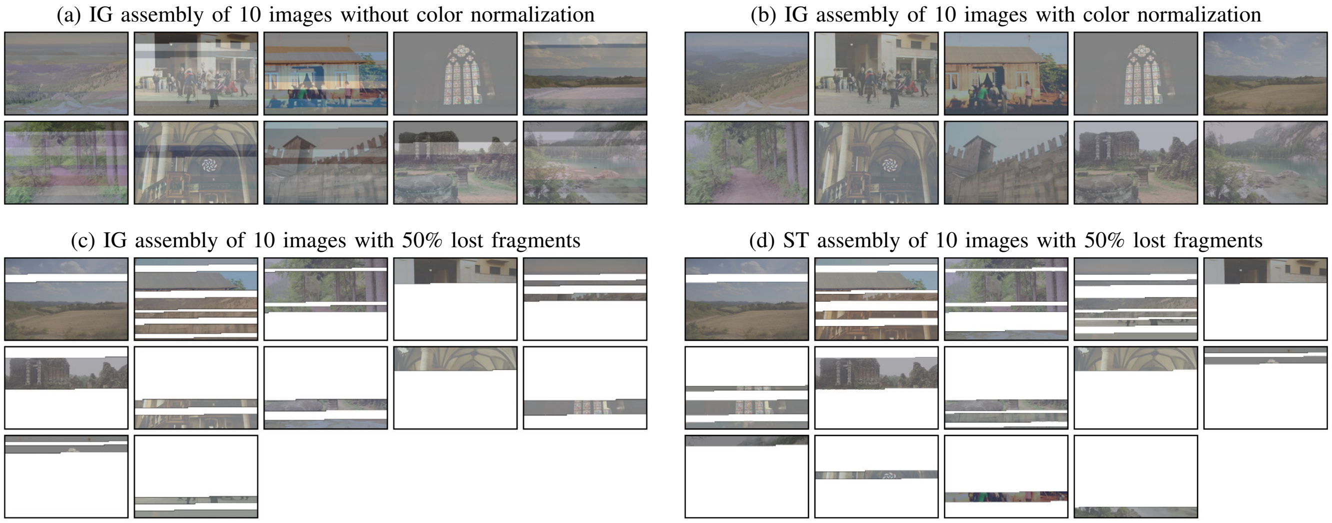
D. Image Assembly Performance

We evaluated image assembly performance on random configurations of simulated fragments sub-selected from the prepared dataset. In each iteration, we chose a fixed number of images in the mixture (10, 25, 50 or 100) and then removed a random fraction of them (uniform selection with drop rate $\rho \in \{0, 0.1, \ldots, 0.5\}$). For both the spanning tree (ST) and the interval graph (IG) assemblers, we swept the decision thresholds to generate empirical trade-off characteristics $r_I(r_C)$ and $r_F(r_C)$. Each drop rate is represented by a different curve. All configurations were repeated 10 times with different images in the mixture and different selection of dropped fragments.

a) Quantitative Performance: The obtained results are shown in Fig. 10. The left column shows the trade-off between the number of found images and the number of identified clusters. The middle column shows an analogous trade-off for the number of correctly placed fragments. The plots reveal similar behavior with perfect assembly when all fragments are available. As more fragments go missing, performance gradually deteriorates. The plots show results for 100 images in the mixture. For smaller number of images, the results are better but reveal the same qualitative behavior. Even for cases with missing fragments, the algorithms allow to assemble a lot of potentially useful evidence.

For easier comparison of the ST and IG assemblers, the right column in Fig. 10 shows the number of correctly placed fragments (top) and the typical runtime of the algorithms (bottom). Both algorithms were implemented in Matlab and used the same functions for computing fragment compatibility (the most time-consuming part of the assembly process). The observed variation of runtime stemmed from various compatibility thresholds and random selection of images/fragments in each evaluated case.

It can be observed that the IG assembler delivers competitive assembly performance with significantly lower runtime. The ST assembler was evaluated in two configurations with full evaluation of the complete affinity matrix and with only a small number of configurations around the diagonal (see Sec-


(a) IG assembly of 10 images without color normalization

(b) IG assembly of 10 images with color normalization

(c) IG assembly of 10 images with 50% lost fragments

(d) ST assembly of 10 images with 50% lost fragments

Fig. 11. Evidence assembly examples for the proposed IG assembler: (a)-(b) assembly of 10 images from 103 orphaned fragments without missing pieces; (c) assembly from 53 fragments (50% missing); (d) ST assembly of the same fragments as in (c) shown for reference.

tion III-F). Despite considerable improvement, the proposed IG solver remains 2-3 times faster.

b) Qualitative Examples: Fig. 11 shows examples of images assembled by the proposed IG algorithm. Examples in (a) and (b) illustrate assembly of 10 images out of 103 mixed and distorted fragments. The results are shown without and with the final brightness and color normalization. Example (c) shows the same 10 images reconstructed from 53 randomly chosen fragments. It can be observed that the algorithm can correctly stitch fragments separated by small holes. Despite imperfect reconstruction, the assembled images represent meaningful content, upon which further actions can be taken. For reference, (d) shows the corresponding reconstruction result for the baseline ST assembler.

A large reconstruction example is shown in Fig. 12. The presented case included 100 images in the mixture and involved: (a) 1092 orphaned fragments; (b) and (c) 636 randomly selected fragments (40% missing). If all fragments are available, it is possible to perfectly reconstruct the input images. In case of missing fragments, holes in the images often inhibit fragment matching. A compatibility threshold can then be used to control the trade-off between the number of discovered clusters and their purity. The examples in (b) and (c) correspond to a strict and a lenient threshold for the proposed IG assembler. The corresponding results for the baseline ST assembler are shown in (d) and (e). For the sake of presentation clarity, in both cases, the results are ordered by cluster size and coherence.

c) Assembling Near Duplicates: In this experiment, we selected a sub-set of 50 similar images, including both semantically similar photos as well as near duplicates (Fig. 13a). We then repeated the same evaluation as before by randomly dividing the images into fragments, simulating content distortion and removing various fractions of the fragments. Just as before, the experiment was repeated 10 times. In this case, we evaluated only the proposed IG assembler.

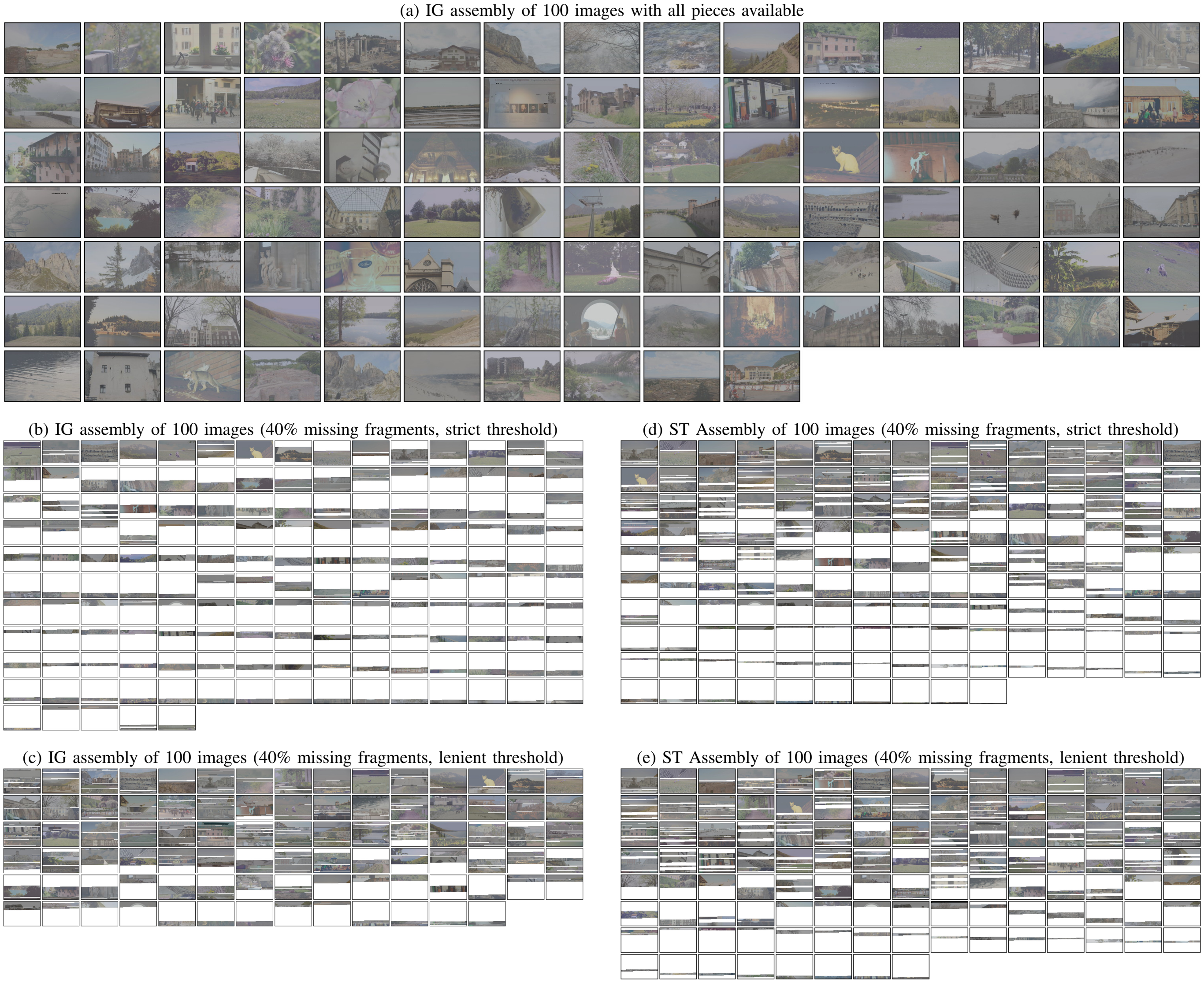
The obtained quantitative results are collected in Fig. 13b. While the performance is slightly worse than for the general case, our algorithm can still extract considerable amount of useful evidence. In case all fragments are available, the reconstruction is nearly perfect. Manual inspection revealed only a few mistakes, which happened when the truncation points coincided by chance and there was no visual content to differentiate between candidate matches. In case of missing pieces, our IG assembler is still capable of delivering useful results. As an example, we've included top 15 images recovered by the IG assembler from an example experiment with fragment drop rate $\rho = 0.4$. For the sake of presentation clarity, we have manually boosted the contrast of the images.

V. Practical Applicability

The proposed algorithm relies on digital image forensics to facilitate JPEG photo carving in the most challenging conditions. Our solution requires PRNU fingerprints of the cameras that were used to take the carved photographs. The fingerprints are used to group image fragments by the source camera and to precisely locate image fragments within the original frame of the sensor. Additional clues, like the image width, may be used to speed-up the grouping.

While the PRNU fingerprint is not always readily available, it can be estimated from existing uncorrupted images. The images may come from an independent storage, or even be obtained in the process of carving the investigated drive assuming enough undamaged photographs can be extracted by conventional means. If the involved camera is available, an arbitrarily high-quality fingerprint can be obtained from new photographs. The minimum number of images needed for a good fingerprint depends on their content, and typically varies between 30 (for favorable bright and low-texture photographs) and 60 (for arbitrary natural photographs).

The carved fragments do not need to come from pristine images. Both local retouching (e.g., red eye, or blemish removal) and global post-processing (e.g., brightness or contrast changes) are allowed, since PRNU matching is robust



(a) IG assembly of 100 images with all pieces available

(b) IG assembly of 100 images (40% missing fragments, strict threshold)

(d) ST Assembly of 100 images (40% missing fragments, strict threshold)

(c) IG assembly of 100 images (40% missing fragments, lenient threshold)

(e) ST Assembly of 100 images (40% missing fragments, lenient threshold)

Fig. 12. Assembly example for a large-scale problem: (a) 100 images reconstructed from 1092 fragments (no missing pieces); (b)-(c) 100 images reconstructed from 636 fragments (40% pieces missing) by the proposed IG assembler; (d)-(e) the same problem reconstructed using the baseline ST assembler. A high resolution version of these results is available in supplementary materials.

enough to work in such conditions. Resizing and cropping are currently not supported, since they destroy synchronization with the camera fingerprint, and a brute-force search would result in prohibitive computational complexity.

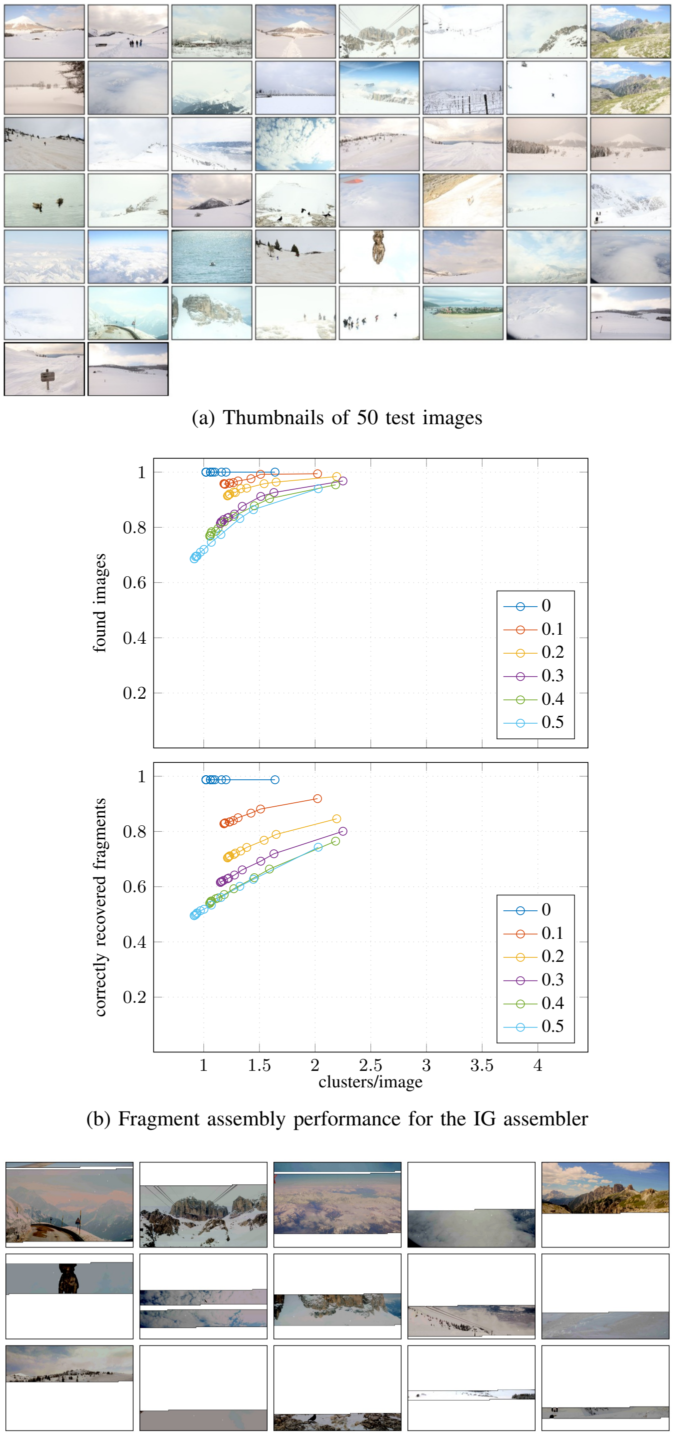
VI. Conclusions

Our study addressed the problem of assembling useful images from a novel source of forensic evidence - orphaned photo fragments. Such fragments are carved without access to file meta-data and are therefore susceptible to reconstruction artifacts including brightness and color normalization problems. These artifacts render existing methods ineffective and make it difficult to match even directly adjoining fragments.

The above problems lead to a novel variant of the jigsaw puzzle problem which involves assembling such orphaned fragments into meaningful images that could be used in law enforcement and data recovery practice. The considered puzzle is one of the most challenging variants of the problem as it involves dealing with missing pieces and mixtures of unknown numbers of images. Dealing with colorization artifacts is necessary both when determining fragment compatibility and when stitching the fragments into the final images.

For the discussed problem, we have exploited forensic image analysis which can provide clues about the location of the fragments within the original image frame. We have proposed an efficient assembly algorithm based on interval graph theory. The algorithm delivers comparable assembly performance to state of the art puzzle solvers based on a minimal spanning tree formulation. However, it requires fewer computations of fragment compatibility - the most time-consuming step of image assembly - which translated in 2-3 times lower runtimes with our reference Matlab implementations.

We believe the described research makes an important step forward towards adoption of orphaned photo carving in regular forensic and data recovery practice. The problem is particularly important for high-profile and sensitive child



(a) Thumbnails of 50 test images

(b) Fragment assembly performance for the IG assembler

(c) Best 15 clusters identified by the IG assembler when drop rate $\rho = 0.4$

Fig. 13. Fragment assembling performance for a sub-set of similar images. The input images in (a) are shown in their original form. The reconstruction results in (c) have boosted contrast for better presentation clarity.

sexual abuse cases, where every piece of evidence may lead not only to the conviction of the perpetrators but also to reaching and helping the victims.

Acknowledgements

We would like to thank E. Uzun and H. Sencar for sharing their prototype of the orphaned photo carver, and anonymous reviewers for helpful comments.

References

[1] G. Richard and V. Roussev, "Scalpel: A frugal, high performance file carver.," in *DFRWS*, 2005.

[2] A. Pal and N. Memon, "The evolution of file carving," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 59–71, March 2009.

[3] E. Chan, S. Venkataraman, F. David, A. Chaugule, and R. Campbell, "Forenscope: A framework for live forensics," in *Proc. Annual Computer Security Applications Conf.*, New York, NY, USA, 2010, ACSAC '10, pp. 307–316, ACM.

[4] J. Sylve, A. Case, L. Marziale, and G. Richard, "Acquisition and analysis of volatile memory from android devices," *Digital Investigation*, vol. 8, no. 3-4, pp. 175–184, 2012.

[5] T. Van Deursen, S. Mauw, and S. Radomirović, "mcarve: Carving attributed dump sets," in *Proc. USENIX Conference on Security*, Berkeley, CA, USA, 2011, SEC'11, pp. 8–8, USENIX Association.

[6] B. Saltaformaggio, R. Bhatia, Z. Gu, X. Zhang, and D. Xu, "Guitar: Piecing together android app guis from memory images," in *Proc. ACM SIGSAC Conf. on Computer and Communications Security*, New York, NY, USA, 2015, CCS '15, pp. 120–132, ACM.

[7] B. Saltaformaggio, R. Bhatia, Z. Gu, X. Zhang, and D. Xu, "Vcr: App-agnostic recovery of photographic evidence from android device memory images," in *Proc. ACM SIGSAC Conf. on Computer and Communications Security*, New York, NY, USA, 2015, CCS '15, pp. 146–157, ACM.

[8] G. Bonetti, M. Viglione, A. Frossi, F. Maggi, and S. Zanero, "A comprehensive black-box methodology for testing the forensic characteristics of solid-state drives," in *Proc. Annual Computer Security Applications Conf.*, New York, NY, USA, 2013, ACSAC '13, pp. 269–278, ACM.

[9] A. Pal, H. Sencar, and N. Memon, "Detecting file fragmentation point using sequential hypothesis testing," *Digit. Investig.*, vol. 5, pp. S2–S13, Sept. 2008.

[10] J. De Bock and P. De Smet, "Jpgcarve: An advanced tool for automated recovery of fragmented jpeg files," *IEEE Tran. Inf. Forensics Security*, vol. 11, no. 1, pp. 19–34, 2016.

[11] E. Uzun and H. T. Sencar, "Carving orphaned jpeg file fragments," *IEEE Tran. Inf. Forensics and Security*, vol. 10, no. 8, pp. 1549–1563, 2015.

[12] E. Durmus, M. Mohanty, S. Taspinar, E. Uzun, and N. Memon, "Image carving with missing headers and missing fragments," in *Proc. IEEE Int. Workshop Information Forensics Security*, 2017.

[13] S. Garfinkel, "Carving contiguous and fragmented files with fast object validation," *digital investigation*, vol. 4, pp. 2–12, 2007.

[14] N. Memon and A. Pal, "Automated reassembly of file fragmented images using greedy algorithms," *IEEE Tran. Image Process.*, vol. 15, no. 2, pp. 385–393, 2006.

[15] H. Liu, S. Cao, and S. Yan, "Automated assembly of shredded pieces from multiple photos," *IEEE Tran. Multimedia*, vol. 13, no. 5, pp. 1154–1162, 2011.

[16] F. Richter, C. Ries, N. Cebron, and R. Lienhart, "Learning to reassemble shredded documents," *IEEE Tran. multimedia*, vol. 15, no. 3, pp. 582–593, 2013.

[17] D. Pomeranz, M Shemesh, and O. Ben-Shahar, "A fully automated greedy square jigsaw puzzle solver," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2011.

[18] G. Paikin and A. Tal, "Solving multiple square jigsaw puzzles with missing pieces," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 4832–4839.

[19] K. Son, D. Moreno, J. Hays, and D. B. Cooper, "Solving small-piece jigsaw puzzles by growing consensus," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 1193–1201.

[20] A. Gallagher, "Jigsaw puzzles with pieces of unknown orientation," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2012, pp. 382–389.

[21] M. Sagiroglu and A. Erçil, "A texture based matching approach for automated assembly of puzzles," in *IEEE Int. Conf. Pattern Recognition*, 2006, vol. 3, pp. 1036–1041.

[22] N. Alajlan, "Solving square jigsaw puzzles using dynamic programming and the hungarian procedure," *American Journal of Applied Sciences*, vol. 6, no. 11, pp. 1941, 2009.

[23] T. S. Cho, S. Avidan, and W. T. Freeman, "A probabilistic image jigsaw puzzle solver," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 183–190.

[24] M. Chen, J. Fridrich, M. Goljan, and J. Lukás, "Determining image origin and integrity using sensor noise," *IEEE Tran. Inf. Forensics Security*, vol. 3, no. 1, pp. 74–90, 2008.

[25] M. Kharrazi, H. Sencar, and N. Memon, "Blind source camera identification," in *ICIP*, 2004, pp. 709–712.

[26] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Tran. Inf. Forensics Security*, vol. 1, no. 2, pp. 205–214, 2006.

[27] S. Masuda, K. Nakajima, T. Kashiwabara, and T. Fujisawa, *Efficient Enumeration of Maximal and Maximum Independent Sets of an Interval Graph and a Circular-arc Graph*, Number 1882 in Computer science technical report series. University of Maryland, 1987.

[28] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva, "Blind prnu-based image clustering for source identification," *IEEE Tran. Inf. Forensics and Security*, vol. 12, no. 9, pp. 2197–2211, 2017.

[29] M. Goljan, "Digital camera identification from images–estimating false acceptance probability," in *Int. Workshop on Digital Watermarking*. Springer, 2008, pp. 454–468.

[30] J. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. American Mathematical society*, vol. 7, no. 1, pp. 48–50, 1956.

[31] D. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "RAISE - a raw images dataset for digital image forensics," in *Proc. of ACM Multimedia Systems*, 2015.

[32] "DDE laboratory," http://dde.binghamton.edu/, visited Sept. 2015.

Emre Durmus received the B.Eng. degree in industrial engineering from Koç University and the M.A. degree in computer science from Brooklyn College. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, New York University Tandon School of Engineering. His research interests include digital forensics, biometrics, authentication, and machine learning.

Paweł Korus received his M.Sc. and Ph.D. degrees in telecommunications (with honors) from the AGH University of Science and Technology in 2008, and in 2013, respectively. Since 2014 he has been an assistant professor with the Department of Telecommunications, AGH University of Science and Technology, Krakow, Poland. He did his postdoctoral research at the College of Information Engineering, Shenzhen University, China. He is currently a visiting professor at the Tandon School of Engineering, New York University, USA.

His research interests include various aspects of multimedia security, image processing, and low-level vision, with particular focus on content authentication and protection techniques for digital photographs. In 2015 he received a scholarship for outstanding young scientists from the Polish Ministry of Science and Higher Education.

Nasir Memon is a professor in the Department of Computer Science and Engineering at NYU Tandon School of Engineering. He is one of the founding members of the Center for Cyber Security (CCS) , a collaborative initiative of multiple schools within NYU including. He is the founder of CSAW, and the Offensive Security, Incident Response and Internet Security Laboratory (OSIRIS) lab at NYU Tandon. His research interests include digital forensics, biometrics, data compression, network security and human behavior. Memon earned a Bachelor of Engineering in Chemical Engineering and a Master of Science in Mathematics from Birla Institute of Technology and Science (BITS) in Pilani, India. He received a PhD in Computer Science from the University of Nebraska.

Professor Memon has published over 250 articles in journals and conference proceedings and holds a dozen patents in image compression and security. He has won several awards including the Jacobs Excellence in Education award and several best paper awards. He has been on the editorial boards of several journals and was the Editor-In-Chief of Transactions on Information Security and Forensics. He is an IEEE Fellow and an SPIE fellow.