Robust Key-point based Data Hiding

Anindya Sarkar, Lakshmanan Nataraj and B. S. Manjunath,
Department of Electrical and Computer Engineering,
University of California, Santa Barbara

1 Problem Statement

We propose a robust data-hiding method which can survive a host of global, geometric and image editing attacks. Localized redundant embedding around multiple key-point centered regions are used to provide robustness to local attacks and geometrical transformations. The primary contributions of the work include novel methods for introducing synchronization information that can be recovered at the decoder without any side information. This synchronization information is then used to identify accurately the geometrical transformation. We show through experiments that our approach is robust to many standard image processing attacks, including JPEG type compression. Quantization index modulation based embedding and repeat accumulate code based error correction are used to obtain a good trade-off between hiding rate and error resilience. Our hiding scheme is robust against rotation angles within $(-45^\circ, 45^\circ)$ and scaling factors within (0.5,2). Detailed experimental results are provided.

In this technical report, we focus on the following aspects of the robust data hiding framework.

1. For geometric synchronization, we introduce some peaks in the frequency domain (DFT magnitude domain) for the input image and the decoder's goal is to properly identify these peaks from the frequency domain representation of the received image - we refer to these peaks as the template peaks. Based on the two sets of matched peaks, we can compute the transformation matrix between the frequency domain representations of the original and received images. In [1, 2] it has been shown that if the transformation matrix (A_{DFT}) is known in the DFT domain, the corresponding matrix in the pixel domain (A) can be computed using the relation: $A = ({A_{DFT}}^{-1})^T$, as shown in Sec. 2. A challenge in using the DFT peaks is that JPEG compression introduces spurious peaks for every inserted DFT peak. We further assume that we know the correspondence between the peaks introduced at the encoder side and those at the decoder, i.e. peaks introduced in a certain quadrant do not get shifted to a different quadrant after the transformation. With this assumption, the problem reduces to just identifying the correct DFT peaks per quadrant.

2. The embedding distortion introduced by the DFT peaks results in striations in the pixel domain. As the strength of the DFT peaks is increased, the peaks are easier to detect; however, the image stria-

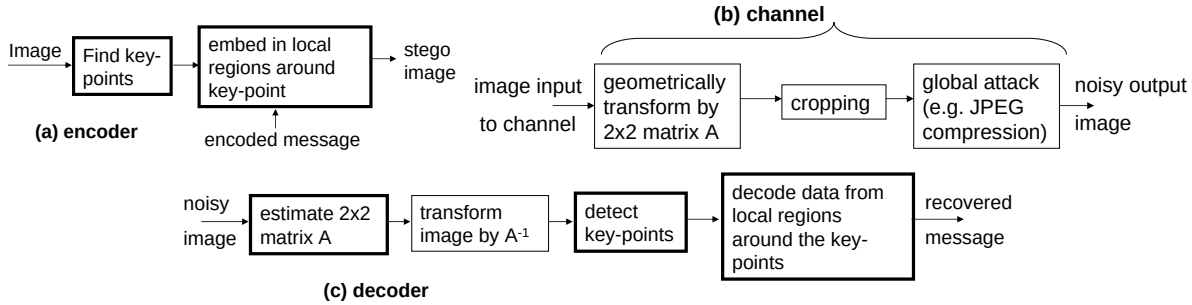


Figure 1: The end-to-end data hiding framework (a) encoder: the same encoded sequence is embedded in every local region around a key-point, (b) channel: the stego image can be subjected to geometrical transformations, local and global attacks, (c) decoder: it aligns the received image with the original image grid and decodes data from the local regions around detected key-points. The boxes outlined in bold represent the main challenges/tasks at the encoder and the decoder.

tions become more visible. We show illustrative examples in Sec. 3 as to how the perceptual quality (visibility of the striations) varies as the strength of the DFT peaks increases.

3. The size of the DFT grid that is considered before peak insertion is also important. For a bigger sized DFT grid, the search process for the DFT peaks becomes more computationally intensive. On the other hand, the actual locations of the DFT peaks can be identified more precisely for a larger sized DFT grid. We study this trade-off in Sec. 4 and observe how much the detection (of geometric transformation parameters) accuracy increases using a DFT grid size that exceeds 512×512 (since most input images are originally 512×512 in size, we use a DFT grid of the same size).

4. The hiding system is robust to cropping as the hiding is repeated in multiple local regions (Sec. 5). It is also possible to recover the geometric transformation exactly provided that the cropping is small enough - cropping in the pixel-domain corresponds to a smoothing of the image DFT leading to the peak detection becoming more difficult because of the smoothing.

5. For local geometric transformations, it is difficult to obtain the transformation matrix if various local regions are subjected to different transformations, after the entire image has been subjected to a certain transformation. In Sec. 6, we observe that if these local regions are small enough, the global transformation matrix can be recovered properly. Thus, we are able to properly align the received image with the original image, except for those regions which are subject to local transformations. Provided that there are some embedding regions which are not subjected to local transformation, data recovery is possible in general.

6. In Sec. 7, we show how the different key-point (KP) detectors perform under cropping and different geometric transformations, after the KP pruning algorithm has been applied.

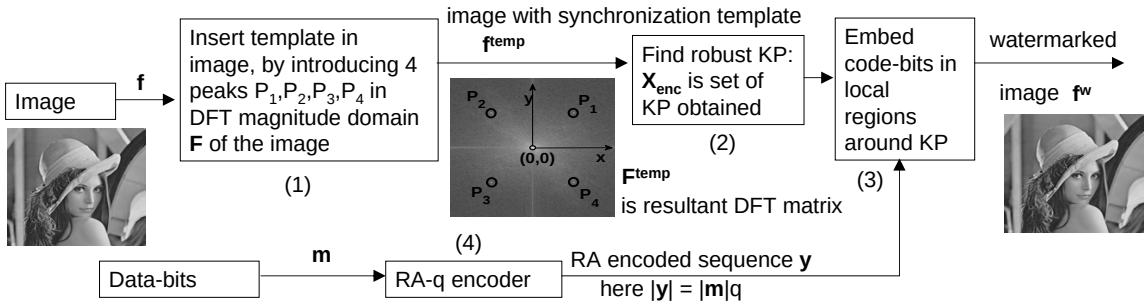


Figure 2: Block diagram at the encoder side where the input image f is transformed to f^w after data embedding - numbers (1)-(4) correspond to the encoder side modules.

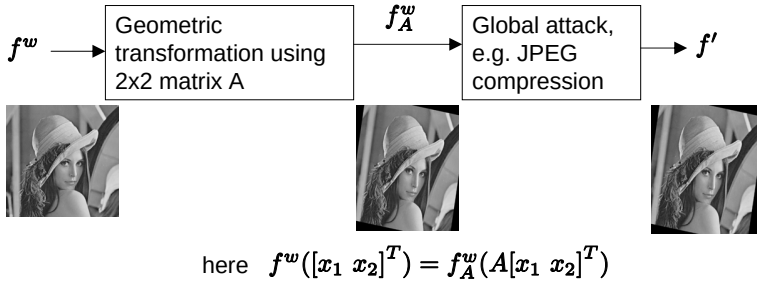


Figure 3: Image modifications (geometric transformation + other attacks) in the channel convert f^w to f'

7. In Sec. 8, We also provide visual examples to demonstrate the relationship that exists between JPEG-induced peak locations with the actual location of the inserted peak.

8. The Photoshop-based image processing filters with their parameter settings used in the experiments are described in detail (Sec. 9). For some local nonlinear transformation based filters, the corresponding change to the DFT magnitude plot is also local and nonlinear leading to difficulty in detection of the actual peaks.

A list of relevant notations is presented in Table 1.

2 Relationship Between the Transformation Matrices in Spatial and DFT domains

We show that a simple relationship exists between the spatial-domain transformation matrix and the corresponding matrix in the DFT magnitude domain - the relation is mentioned in [2][1] and we include the proof here for completeness and ease of understanding. The transformation is computed w.r.t. the image center in the pixel domain and w.r.t. the center of the DFT grid in the DFT domain.

Table 1: **Glossary of Notations**

Notation	Definition
$f/f^{temp}/f^w/f'$	f: Original image/ f^{temp}: image after synchronization template addition/ f^w: watermarked image/ f': output image from the channel, after noise attacks on f^w
$F/F^{temp}/F^w/F'$	the DFT matrix corresponding to $f/f^{temp}/f^w/f'$, respectively
F_{mag}/F_{phase}	F_{mag}: the magnitude component of the DFT matrix F, F_{phase}: the phase component of F
$\{P_1, \cdots, P_4\}$	location of DFT domain peaks added to F to produce F^{temp} - inserted as self-synchronization template
$\{Q_1, \cdots, Q_4\}$	location of peaks extracted from the DFT magnitude plot of f', i.e. F' - they are integer valued points
$\{R_1, \cdots, R_4\}$	the ideal peak locations $\{P_1, \cdots, P_4\}$ get mapped to $\{R_1, \cdots, R_4\}$ (real numbers) in the DFT plot of the received image - if the geometric transformation is estimated "correctly enough", Q_i = rounded version of(R_i), $1 \leq i \leq 4$
X_{enc}	set of key-points obtained from f^{temp} - hiding occurs in $B \times B$ local regions around the key-points
X_{dec}	set of K_{dec} key-points obtained after geometrically aligning f', the noisy received image
f_A	image obtained after geometric transformation of f using $A \in \mathbb{R}^{2\times 2}$: $f_A\left(A[x_1\ x_2]^T\right) = f\left([x_1\ x_2]^T\right)$
A_{DFT}	if A is the geometric transformation between images f^1 and f^2, A_{DFT} is the transformation between DFT plots F^1 and F^2, i.e. $F^2_{A_{DFT}}(A_{DFT}[u_1\ u_2]^T) = F^1([u_1\ u_2]^T) \iff f^2_A(A[x_1\ x_2]^T) = f^1([x_1\ x_2]^T)$
B	the size of a local region used for hiding is $B \times B$
δ_{th}	a threshold imposed on the corner strength for key-point selection while hiding
QF_h	design quality factor (QF) used for hiding
QF_a	output JPEG QF at which the noisy output image f' is advertised
λ	the first λ AC DCT coefficients obtained after zigzag scan are used for embedding for a 8×8 block

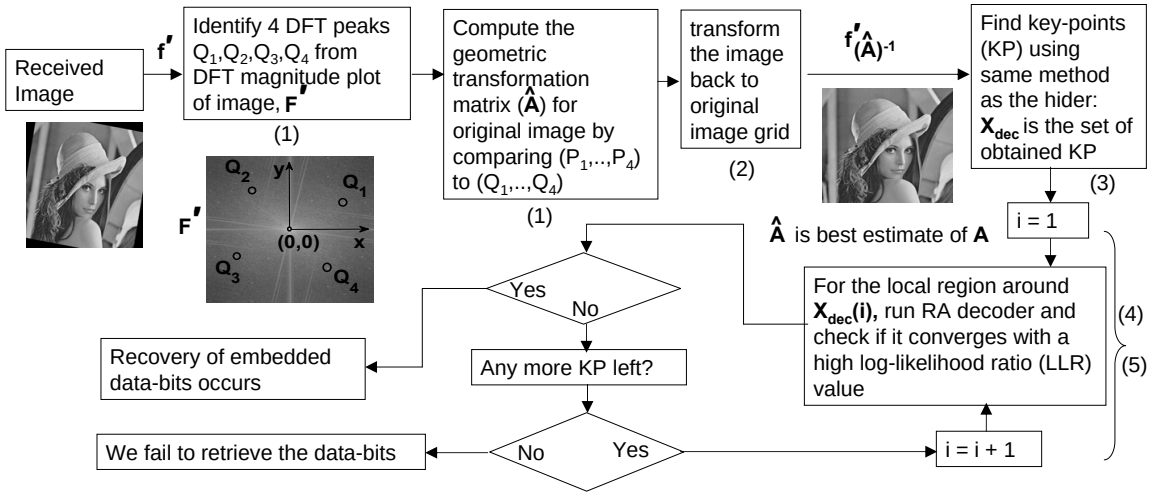


Figure 4: Block diagram at the decoder side where we aim to retrieve the embedded data-bits from f' - numbers (1)-(5) correspond to the decoder side modules.

The synchronization template consists of 4 peaks inserted in the DFT magnitude domain of the $N_1 \times N_2$ input image - we refer to the image with the inserted template as u. After transforming the image using $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$, we obtain u_A where the relation between the pixel locations in u and u_A is:

$$u([x_1\ x_2]^T) = u_A(A[x_1\ x_2]^T),$$

i.e. the point (x_1, x_2) in the image u gets mapped to $(a_{11}x_1 + a_{12}x_2, a_{21}x_1 + a_{22}x_2)$ in u_A.

Let U and U_A denote the 2D $N \times N$ DFT for u and u_A, respectively, where $N = \max(N_1, N_2)$.

$$U(k_1, k_2) = \sum_{x_1} \sum_{x_2} u(x_1, x_2) e^{-2j\pi k_1 x_1/N} e^{-2j\pi k_2 x_2/N},\ 0 \le k_1, k_2 < N$$

$$U_A(k_1, k_2) = \sum_{x_1} \sum_{x_2} u_A(x_1, x_2) e^{-2j\pi k_1 x_1/N} e^{-2j\pi k_2 x_2/N},\ 0 \le k_1, k_2 < N$$

The problem is to express A_{DFT} in terms of A, where $[k_1\ k_2]^T$ in U gets mapped to $A_{DFT}[k_1\ k_2]^T$ in U_A (notation wise, $U(k_1, k_2)$ is equivalent to $U([k_1\ k_2]^T)$).

$$U([k_1\ k_2]^T) = U_A(A_{DFT}[k_1\ k_2]^T) \tag{1}$$

$$u(x_1, x_2) = u_A(a_{11}x_1 + a_{12}x_2, a_{21}x_1 + a_{22}x_2) \Rightarrow$$
$$u_A(x_1, x_2) = u((a_{22}x_1 - a_{12}x_2)/D, (-a_{21}x_1 + a_{11}x_2)/D), \text{ where } D = (a_{11}a_{22} - a_{12}a_{21})$$
$$\therefore\ U_A(k_1, k_2) = \sum_{x_1}\sum_{x_2} u((a_{22}x_1 - a_{12}x_2)/D, (-a_{21}x_1 + a_{11}x_2)/D\ )e^{-2j\pi k_1 x_1/N} e^{-2j\pi k_2 x_2/N}$$
$$\text{Replacing } y_1 = (a_{22}x_1 - a_{12}x_2)/D \text{ and } y_2 = (-a_{21}x_1 + a_{11}x_2)/D, \text{ we get}$$
$$U_A(k_1, k_2) = \sum_{y_1}\sum_{y_2} u(y_1, y_2)e^{-j2\pi y_1(a_{11}k_1 + a_{21}k_2)/N} e^{-j2\pi y_2(a_{12}k_1 + a_{22}k_2)/N} \Longrightarrow$$
$$U_A(k_1, k_2) = U(k_1a_{11} + k_2a_{21}, k_1a_{12} + k_2a_{22}) \Longrightarrow$$
$$U(k_1, k_2) = U_A((a_{22}k_1 - a_{21}k_2)/D, (-a_{12}k_1 + a_{11}k_2)/D) = U_A((A^{-1})^T[k_1\ k_2]^T) \tag{2}$$
$$\text{Thus, we see that } A_{DFT} = (A^{-1})^T, \text{ from (1) and (2)}$$

Thus, if the transformation matrix is known in the DFT domain (A_{DFT}), it is simple to compute the transformation matrix in the pixel domain (A), for any $A \in \mathbb{R}^{2\times 2}$. The received image f', which has been geometrically transformed by A, can be inverse-transformed so that it ($f'_{A^{-1}}$, as in Fig. 4) corresponds with the original image grid (f or f^{temp} or f^w, as in Fig. 2), based on which hiding was done.

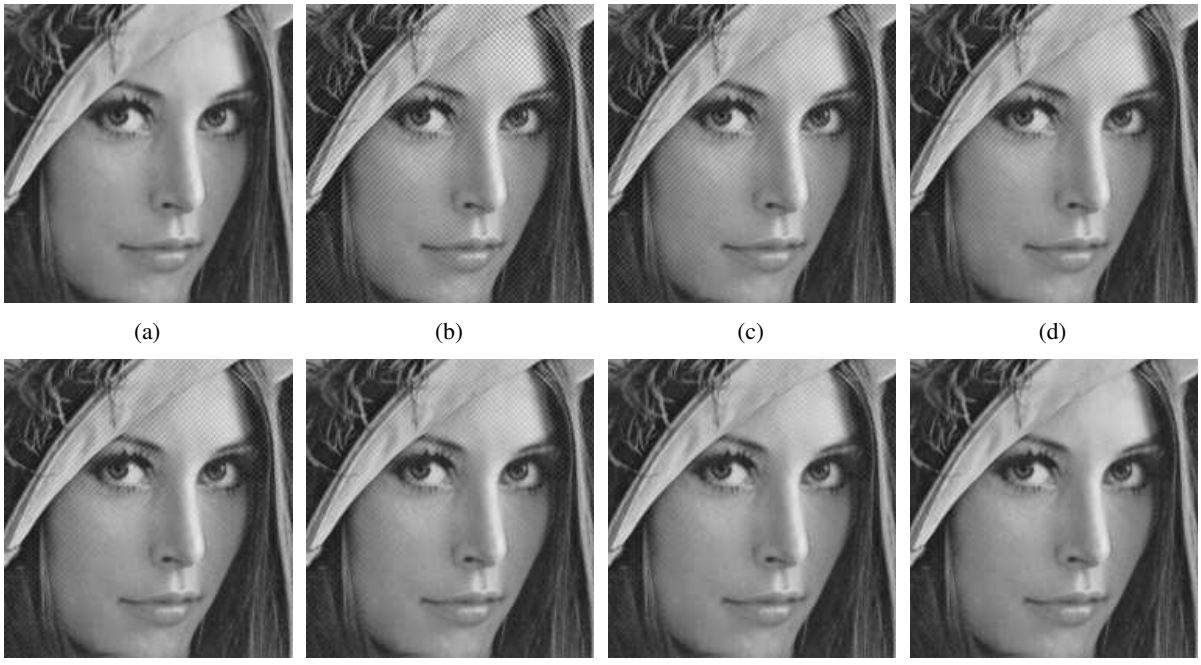
There are 4 parameters to be estimated in A. Each point in the DFT grid is 2-D and hence, 2 points are needed to solve for the 4 unknowns. By conjugate symmetry, once we insert peaks in the first and second quadrants, the corresponding peaks in the third and fourth quadrants get fixed.

3 Illustration of Template Based Embedding Distortion

For more accurate parameter estimation, one can increase the magnitude of the DFT peaks, so that they can be identified more precisely. The trade-off here is that increase in the peak strength may introduce visible periodic striations in the image (Fig. 5).

4 Result on Varying DFT Size in Accuracy of Transformation Estimation

The accuracy of the transformation estimation depends on the resolution we set for the scaling and rotation parameters, as shown in Table 2. Here, we use a DFT size of 512×512 and we assume that the geometric transformation constitutes of rotation and/or scaling (in any sequence). The resolution which we use to compute the angle and scale parameters are denoted by δ_θ and δ_s, respectively. We explain the computation of the probability of cooect estimation of the transformation parameters in A, which is denoted by $p_{A,m}$ in Table 2 for a given set of angle and scale resolutions. E.g. we use a resolution of 0.5° and 0.05 for the angle and scale parameters. At the decoder side, we assume that the computed rotation angle and scale factors are multiples of 0.5° and 0.05, respectively, and the angle and scale values thus computed have to equal 20°



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

(e) (f) (g) (h)

Figure 5: It is seen that the striations become more visible as the PSNR decreases: (a) original image; the other images are obtained after template addition in the DFT domain at different PSNRs (dB) (b) 31.34 (c) 33.46 (d) 36.17 (e) 37.91 (f) 39.68 (g) 41.59 (h) 45.85. The periodic patterns are very clearly evident in the lower PSNR images and the visibility of the periodic patterns progressively decreases with increased PSNR.

and 1.1, respectively, for it to be considered a success in $p_{A,m}$ computation. For all the experiments, we use a rotation angle of 20° and a scaling factor of 1.1 is used along both dimensions. Also, for all subsequent tables (after Table 2), we use $\delta_\theta = 0.5^\circ$ and $\delta_s = 0.05$.

Table 2: $p_{A,m}$ is computed for $\theta = 20^\circ$, $s_x = 1.1$, $s_y = 1.1$ along with 20% cropping, for $QF_a = 75$, and various resolutions for rotation (δ_θ) and scale (δ_s) parameters, e.g. (0.5, 0.05) indicates $\delta_\theta = 0.5^\circ$ and $\delta_s = 0.05$. PSNR values are in dB.

(0.5, 0.05)		(0.25, 0.05)		(1, 0.025)		(0.5, 0.025)		(0.25, 0.025)		(0.125, 0.05)		(1, 0.0125)	
$p_{A,m}$	PSNR	$p_{A,m}$	PSNR	$p_{A,m}$	PSNR	$p_{A,m}$	PSNR	$p_{A,m}$	PSNR	$p_{A,m}$	PSNR	$p_{A,m}$	PSNR
0.94	44.25	0.86	42.81	0.90	43.89	0.90	43.74	0.87	43.22	0.85	42.41	0.74	41.68

We have experimented with the DFT size N, as shown later in Table 3. Since the 250 images we experimented with were 512×512 images, we set N to 512. It is seen that slight performance improvement is obtained using N of 1024 as compared to 512 - the cost involved is searching over $(\frac{1024}{512})^2 = 4$ times as many points compared to when $N = 1024$. Here, we assume that the geometric transformation consists of rotation and/or scaling (in any sequence). We also use the property that the position of the JPEG-induced peaks can be predicted in terms of the actual location of the actual inserted DFT-domain peak. This combination of prior assumption about the transformation and prediction of JPEG peak locations is used for all subsequent experiments in the report.

Table 3: p_A is computed for different DFT sizes:

JPEG QF (QF_h)	DFT size = 512×512		DFT size = 1024×1024	
	Accuracy	PSNR(dB)	Accuracy	PSNR(dB)
40	0.85	40.25	0.86	40.32
50	0.86	43.80	0.87	44.20
60	0.90	45.90	0.92	45.98
75	0.97	47.20	0.98	47.60

5 Effects of Cropping on Accuracy of Geometric Transformation Estimation

We discuss the effect of cropping on the accuracy of the geometric transformation estimation. Cropping can be interpreted as multiplying an image by a rectangle, where the rectangle size determines the image size after cropping. Multiplying by a rectangle in the spatial domain is equivalent to convolving with a 2-D sinc function in the DFT domain. The smaller the size of the rectangle (smaller is the number of pixels retained), the corresponding sinc in the DFT domain will have a wider variance making the DFT of the cropped image more blurred and peak picking more difficult.

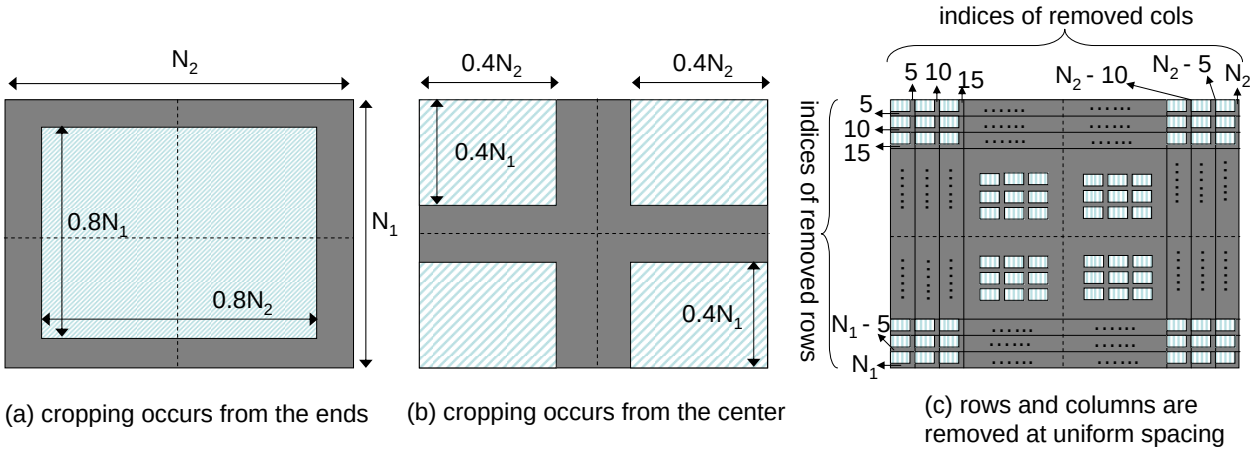


Figure 6: Various cropping approaches for a $N_1 \times N_2$ image - the greyish part denotes the cropped out (discarded) image part.

Three variations of cropping are shown in Fig. 6. Starting off with a $N_1 \times N_2$ image, the cropped image has $0.8N_1$ rows and $0.8N_2$ columns in all the 3 cases (a)-(c). In (a), the discarded pixels come from the ends while in (b), they come from the central part. In (b), the 4 cropped regions are put together to constitute the final image. In (c), 1 row (column) out of every 5 rows (columns) is removed. Thus, on an average, the size of the individual blocks that are retained in the pixel domain is smallest for (c) and hence, the DFT peaks in (c) are maximally blurred by the corresponding higher variance of the sinc functions. Experimental results show that p_A is highest for (a) and lowest for (c).

Table 4: For the cropping experiments, $QF_a = 75$, the starting windowed peak strengths $[v_1\ v_2\ v_3\ v_4] = [14\ 10\ 4\ 2]$, and the results are shown after using various cropping methods - methods (a)-(c) are explained in Fig. 6.

$crop$	Method a		Method b		Method c	
	p_A	PSNR(dB)	p_A	PSNR(dB)	p_A	PSNR(dB)
0.6	0.955	46.90	0.850	44.10	0.518	40.06
0.8	0.970	47.20	0.872	44.33	0.548	42.02

6 Robustness to Small Local Transformations

The template based method is useful for estimation of global affine transformations. If there are small local regions which undergo transformations different from that of the initial global transform, we cannot determine the individual transformations undergone by the local regions. However, if we can still determine

the global transformation in spite of the small local transformations, then the received image can be properly aligned with the original grid and decoding is possible in those image regions which do not suffer local transformations.

When the entire image is transformed using A while there are n small local regions which are transformed using $A_1, A_2, \cdots, A_n$, we can still recover A provided the local regions are small enough. E.g. if we consider four regions in the pixel domain, each centered around a quadrant center, and each region has a different geometrical transformation from that of the overall image, we are still able to recover the global transform if each region is not more than 30% of the quadrant dimensions. In Table 5, we present results for accurate A accurately for a variety of local region sizes. In Table 5, by "quadrant" (0.05), we refer to the case where we take a region of dimension 5% that of the quadrant around the quadrant center, (for all the four quadrants) and then subj3ct it to a transformation, different from the overall global transform (A). By "center"(0.05), we refer to the scenario where a local region of dimension 5% of a quadrant is considered around the DFT center and it is subjected to a transformation different form A.

Effect of Small Local Transformations:

Table 5: Results with small local transformations at varying levels, using QF_a = 75

center	0.05	0.10	0.15	0.20	quadrant	0.05	0.10	0.15	0.20
p_A	0.97	0.96	0.92	0.84	p_A	0.96	0.95	0.91	0.80
PSNR	47.20	46.78	44.88	42.67	PSNR	46.76	44.30	42.10	40.91

7 Comparison of Key-point Detectors

We compare the performance of the various key-point detectors under various attacks (Fig. 7) - it is observed that Nobel-Forstner (NF) key-points result in a higher fr_{match} and also, the embedded databits can be successfully retrieved for a higher fraction of images.

8 Using JPEG Peak Location Property to Discard Noisy Peaks

We make the following observation about the likely position of the JPEG induced peaks relative to an inserted peak location, which is also experimentally validated. If (x, y) corresponds to the geometrically transformed location of the inserted DFT peak, then the JPEG induced peak locations will be at $(x \pm 64k, y)$ and $(x, y \pm 64\ell),\ k, \ell \in \mathbb{Z}$ considering a 512×512 grid. Considering the 30 topmost peaks per quadrant, and comparing the peak locations with the actual location of the inserted DFT peak for geometrically transformed images, we found that 80% of the detected peaks were "JPEG-like neighbors" of the actual peaks. Visual examples of how the JPEG-induced peaks are spatially related with the location of the original DFT peaks are presented in Fig. 8 and 9.

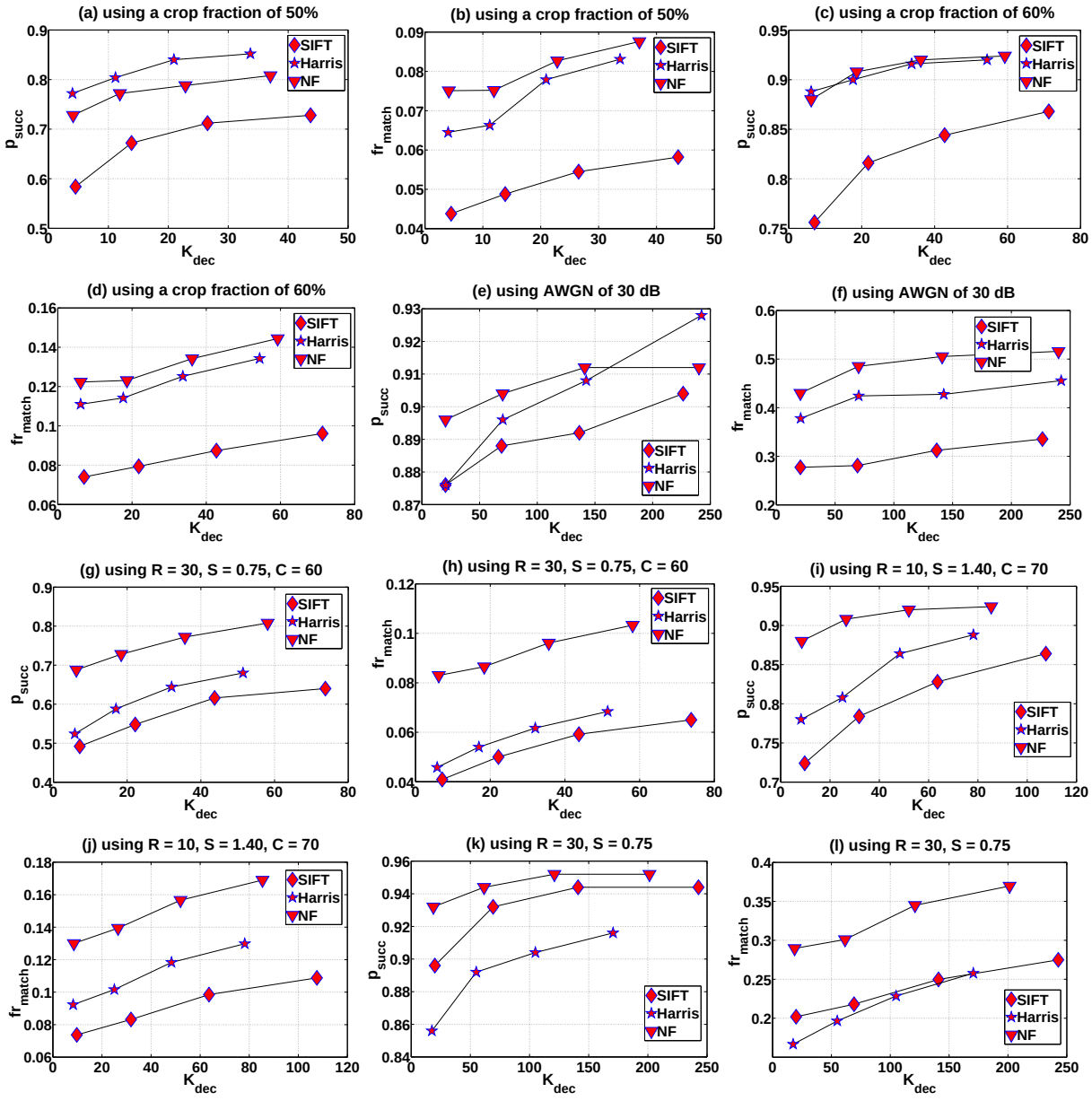


Figure 7: Based on the above experiments, Nobel-Forstner (NF) key-points perform better than Harris and SIFT key-points; here p_{succ} is the fraction of images for which we successfully retrieve the embedded data. The experiments are performed on 250 images and the average fr_{match} is reported. In (g), R=30 refers to a rotation angle of 30°, S=0.75 refers to a scaling factor of 0.75 for both the axes, and C=60 means that after rotation and scaling, the image is cropped while retaining 60% of the image on both axes. In (c)-(d), a crop fraction of 60% means 60% of the image is retained along both the axes.

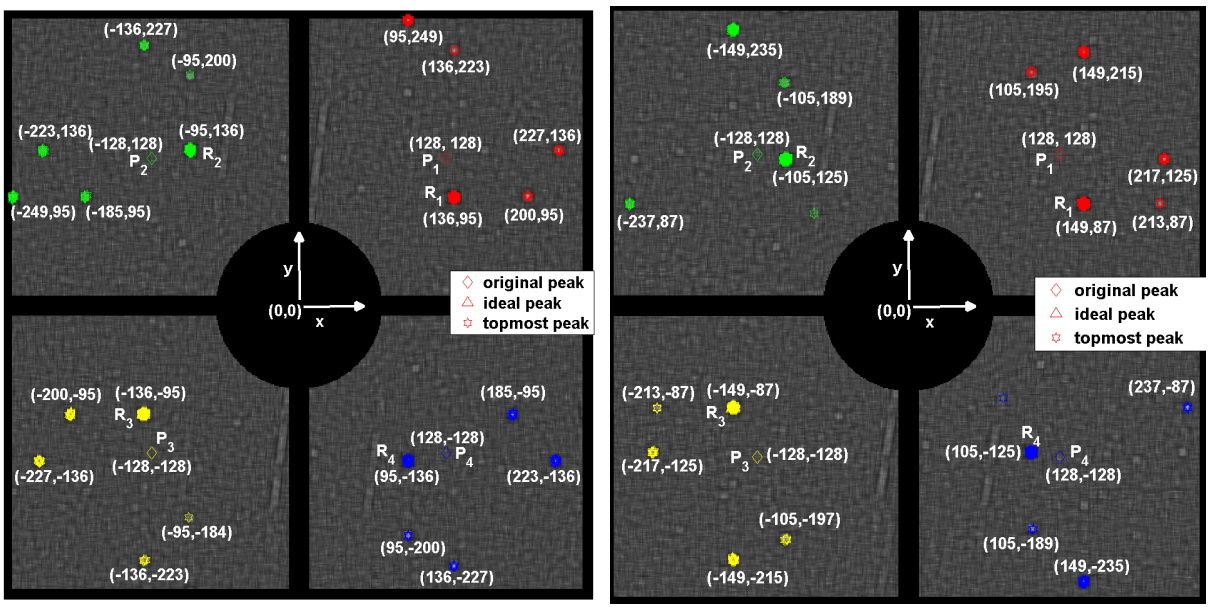


Figure 8: (from left to right) (a) and (b) correspond to 512×512 $\mathbf{T}_\Delta$ plots for transformations of $\{\theta = 10^\circ, s_x = s_y = 1.1\}$ and $\{\theta = 30^\circ, s_x = 1, s_y = 1.2\}$, respectively, along with 20% cropping, and QF_a=75. P_i refers to the original location of peak insertion, which is shifted to R_i (R_i values are rounded here) after the geometric transformation. The 20 topmost peaks are shown per quadrant. Due to the window based peak insertion, many peak locations are clustered together; hence we see fewer peaks per quadrant. Comparing the locations of the topmost peaks with that of the rounded values of R_i, we observe that JPEG-induced peaks are generally separated at multiples of 64 units apart, horizontally and vertically.

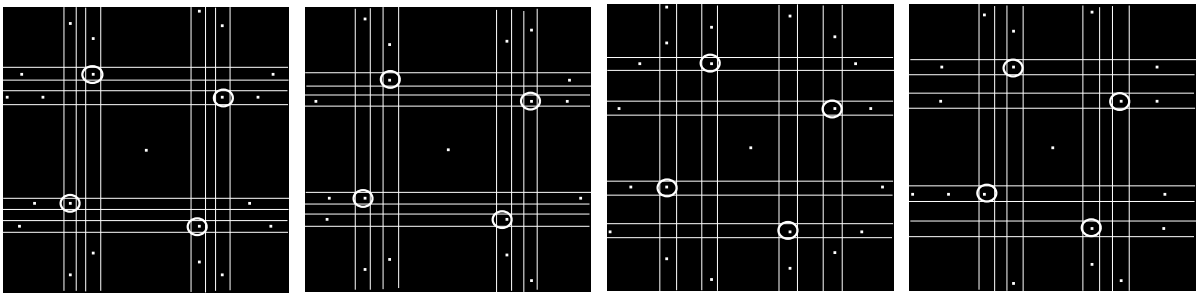


Figure 9: (from left to right) (a)-(d) correspond to $\theta = 10^\circ, s_x = s_y = 1.1$, $\theta = 30^\circ, s_x = 1, s_y = 1.2$, $\theta = 20^\circ, s_x = s_y = 1.1$, and $\theta = 15^\circ, s_x = 1.1, s_y = 1.3$, along with 20% cropping and QF_a = 75. The circled locations denote $\{R_i\}_{i=1}^4$, while the horizontal and vertical lines show how the JPEG-induced peaks (white dots) are at multiples of 64 units apart from $\{R_i\}_{i=1}^4$.

9 Description of Photoshop Attack Parameters

We geometrically transformed 50 images using our default choice of A before subjecting them to different attacks, created using Adobe PhotoShop, and computed the fraction of cases for which we can successfully estimate A. From the figures (Fig. 10), we see that for local non-linear filtering attacks (for attacks like pinch, twirl), the DFT template peaks can no longer be observed in F', the DFT magnitude plot of the received image. This explains why p_A is very low for these filter based attacks. In the future, we will use more attack-specific methods aimed at peak recovery from the DFT of these filtered images.

The parameters used for various Photoshop attacks are as follows:

(i) diffuse glow: graininess=5, glow amount=2, clear amount=20,

(ii) film grain: grain=1, highlight area=0, intensity=1,

(iii) pinch: the pinch factor was varied from 10%-75%,

(iv) spatter: spray radius=1, smoothness=15,

(v) twirl: the twirl angle is varied from 10°-25°,

(vi) unsharp masking: amount=20%, radius=1, pixel threshold=0,

(vii) zigzag: amount=10, ridges=1, style is pond ripples,

(viii) lens blur: iris shape is a hexagon, iris radius=5,

(ix) ocean ripple: ripple size=2, ripple magnitude=2,

(x) dust and scratches: radius=3, threshold=0,

(xi) shear: a list of points is specified and then non-linear distortions are introduced by using splines which pass through these points,

(xii) offset: the horizontal and vertical offsets are 15 and 25, respectively.

References

[1] M. Barni and F. Bartolini, *Watermarking systems engineering: Enabling digital assets security and other applications*. CRC Press, 2004.

[2] R. A. Emmert and C. D. McGillem, "Multitemporal geometric distortion correction utilizing the affine transformation," in *Proc. of IEEE Conf. on Machine Processing Remotely Sensed Data*, Oct 1973, pp. 1B–24 – 1B–32.

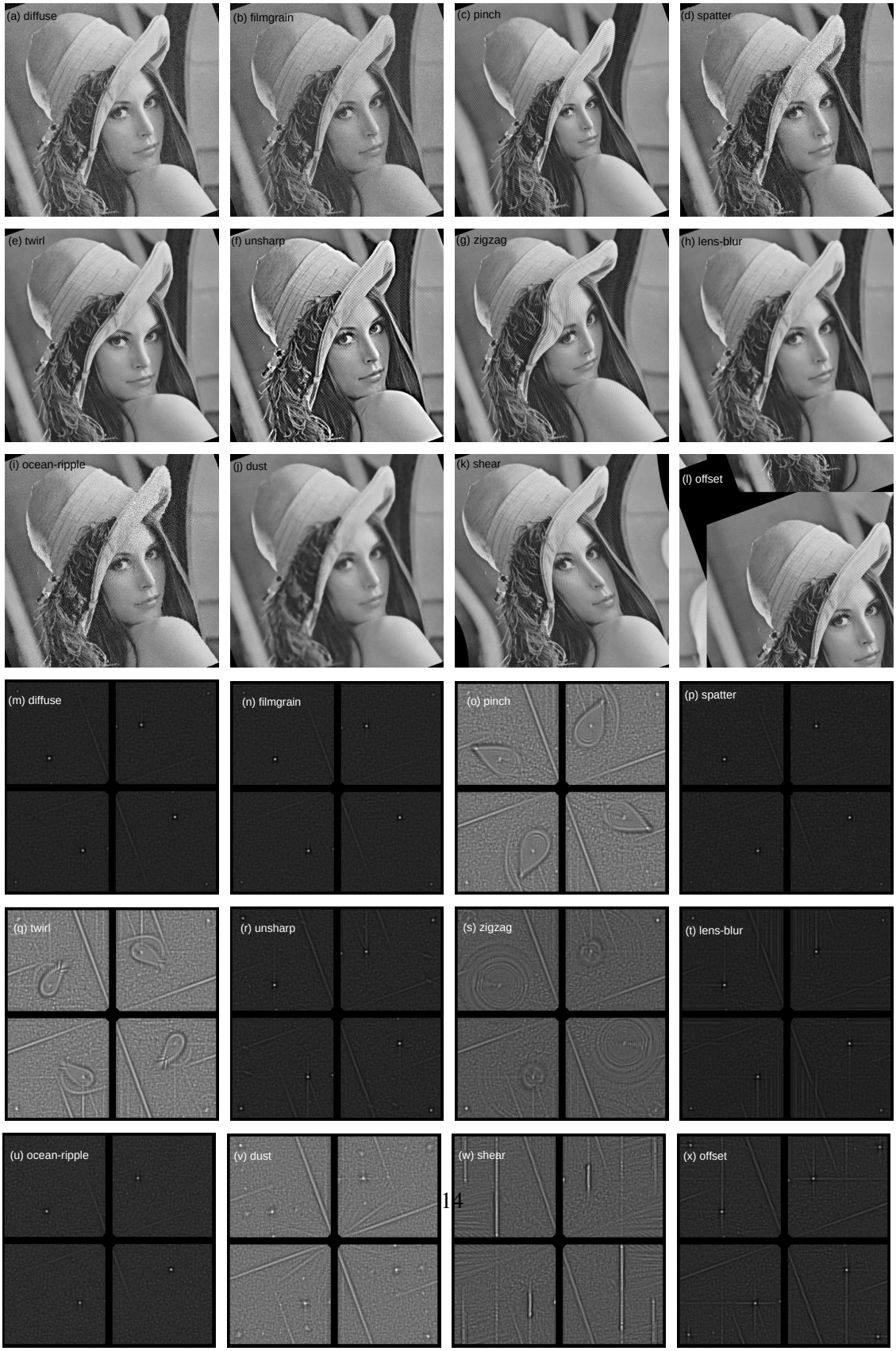


Figure 10: (a)-(l) images and their (m)-(x) DFT magnitude plots after various Photoshop attacks