Morphological Amoeba-based Patches for Exemplar-Based Inpainting

Susana Castillo[1] Douglas W. Cunningham[1] Christian Winger[2] Michael Breuß[3]

[1] BTU Cottbus-Senftenberg
Konrad-Wachsmann-Allee 5,
03046 Cottbus, Germany
castillo@b-tu.de
douglas.cunningham@b-tu.de

[2] Öko-Institut e.V., Germany
Merzhauser Straße 173
79100 Freiburg, Germany
c.winger@oeko.de

[3] BTU Cottbus-Senftenberg
Platz der Deutschen Einheit 1,
03046 Cottbus, Germany
breuss@b-tu.de

ABSTRACT

Inpainting is the process of replacing areas in an image with a perceptually plausible substitution. A common technique is to iteratively match and fill small patches at the edge of the target region making use of similar patches from the same image. Nearly all inpainting algorithms based on this approach use a single patch size for the entire image. Yet, it seems clear that differently sized structures within the same image – for example a leaf versus a car tire – may require different patch sizes in order to achieve reasonable inpainting results. Likewise, a fixed patch size will give different results for the same image when the image resolution is doubled. A reasonable patch should therefore take into account the overall image size as well as the size and shape of the structures at the patch location. The aim of our paper is to study the effect of adaptively altering size and shape of the patch. We show that this technique leads to a better quality of the inpainting result compared to a fixed patch size.

Keywords

Inpainting, Adaptivity, Criminisi algorithm, Morphological Amoeba

1 INTRODUCTION

The class of techniques designed to replace empty regions in an image with perceptually plausible content is called inpainting after Bertalmió et al. [Ber00a]. Inpainting can be used for many purposes in visual computing, including, for example, denoising [Ad17a], image compression [Mai09], or automatic repair of damaged images [Cai17a]. There are many technical approaches to inpainting, cf. [Gui14a] for an overview. These techniques include exploring information from level lines [Mas98a], tackling the task as a texture synthesis problem [Ef99a], or making use of partial differential equations (PDEs) [Ber00a]. One may also combine different techniques, usually with improved results [Ber03a]. Given the wide variety of potential applications and approaches, it is of fundamental interest to explore and understand the different building blocks of the most promising inpainting methods.

One of the most central works in image inpainting is Criminisi et al. [Cri04a] whose algorithm has since become the core of most exemplar-based approaches (see, e.g., [Buy15a] for a broader discussion of the approach). Exemplar-based approaches assume that the best description of the information to be filled in can be found somewhere else in the same image. Exemplar-based inpainting methods follow a general pipeline. First, the border of the empty or target region is located. Second, a pixel on the border is selected and a small patch is centered on the selected pixel. Note that part of the patch will contain valid image information and part will be in the target region. The size of the patch is set manually, with the size usually chosen to match the largest relevant feature in the image. Traditionally, patch sizes of 5×5, 7×7, 9×9 and sometimes 11×11 pixels are used [Lem13a]. The third step, filling-in, is subdivided into finding a matching patch and copying the new patch into the target patch. This process is iterated until all holes have been filled. A number of newer algorithms have altered individual building blocks in Criminisi et al.'s [Cri04a] pipeline. Modifications encompass attempts to produce better descriptions of the contents of a patch (e.g., [Lem11a, Xu10a]), constructing a more efficient matching process (e.g., [Xi13a, Ngu13a]) or proposing more elaborate texture propagation/copy procedures (e.g., [Kom07a, Lem13a]). Nearly all of the proposed

https://doi.org/10.24132/JWSCG.2018.26.2.6

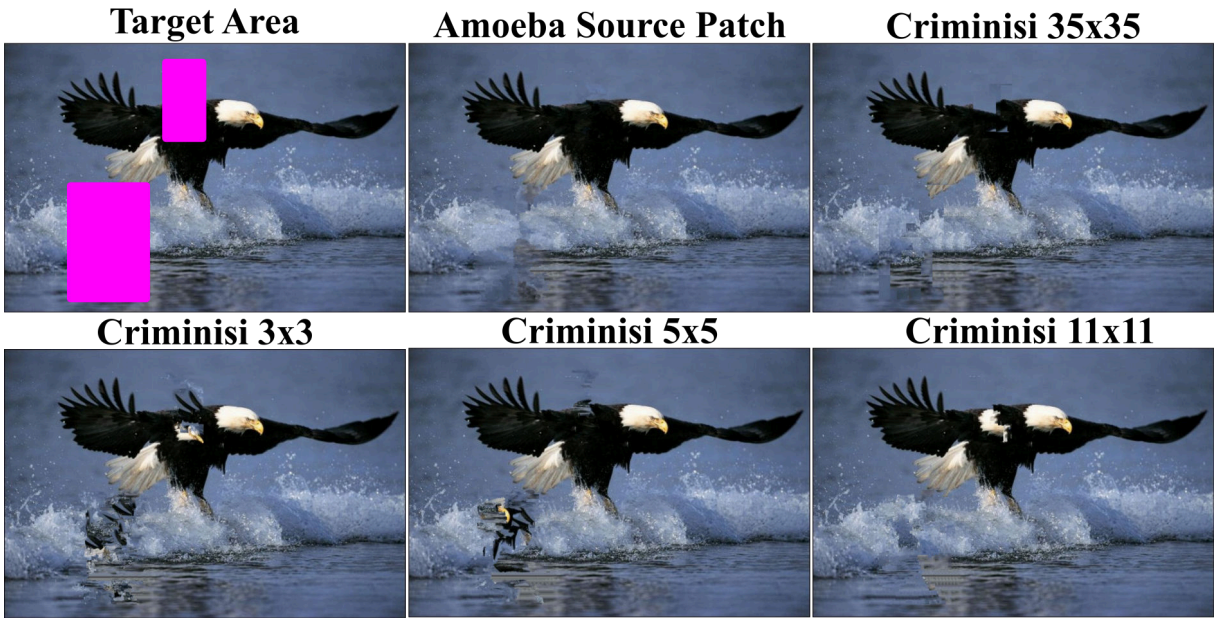


Figure 1: In reading order: Original image to be restored, the result of our new variable patch shape approach, and four results of an exemplar-based method – with two extreme (35×35 and 3×3) and two commonly used (5×5 and 11×11) patch-sizes

improvements leave the underlying patch concept unchanged, using a one-size-fits-all approach to patch size. While one can set the patch size to match image size or general trends in feature size, it appears evident that no uniform patch size and shape can capture the range of possible feature shapes. For example, structures such as long, bold edges, may require large, non-square patches. Structures that rapidly change such as a raged edge would require a smaller patch (see Figure 1).

The most similar work to the approach we will study is the work of Wu and Ruan, who proposed that patch size could be changed dynamically to match local texture information [Wu09a]. As usual, they created a small patch (4×4) around each pixel at the edge of the target region. They then calculated the color variation in each patch allowed to either grow (to 5×5) or shrink (to 3×3) based on a user-defined threshold. They argued that a lot of color variation probably represents a textured region and should therefore be assigned a larger patch in hopes of capturing more of the texture. A homogeneous region, on the other hand, will have no color variation. They claimed that patches for homogeneous regions should be kept small to prevent accidentally introducing structure. After filling-in, a divergence constrained PDE is used to reduce differences between neighboring patches.

In this paper, we propose that patch size should be based on the size and shape of local structures. Large structures should get large patches, small structures should get small patches. Likewise, non-square structures should get non-square patches. Since all previous work on exemplar-based inpainting exclusively used square patches, they generally either copied undesired structure (introducing artifacts) or copied partial structures (creating salient discontinuities). These *intrusion artifacts* can be seen in the fixed-patch size examples in Figure 1.

In **our contribution** we argue that, to completely capture the local structure at the boundary of the target region, the patch size should be iteratively altered until the local structure is fully enclosed. This would require that the method determines when the local structure is fully enclosed. Color variation as considered in [Wu09a] appears not to be able to capture the local image information, as the same variation in a patch's color may arise from a long edge or from a scattered texture. Thus, some form of feature extraction is needed. We follow the image segmentation ideas from [Ler07a] to create – at the source area – flexible, dynamic patches of arbitrary size and shape that capture and therefore copy only relevant structure. This focus on segmenting and copying the relevant structures helps to avoid intrusion artifacts and leads to visually pleasant results (see Figure 1).

2 ALGORITHM

Just like most exemplar-based inpainting techniques, ours is based on Criminisi et al.'s [Cri04a] algorithm, which showed that the order in which the target regions gets filled-in is important. Prioritizing patches in which structural elements are pointing inwards into the target area produces considerably better completions. Criminisi et al.'s algorithm is illustrated in Figure 2 and is

discussed in more detail in the remainder of this section.

In a pre-processing step, the image is converted into CIE L*a*b color space and all operations are performed on the three color channels simultaneously. In the first step, a target pixel located along the border of the target region $\partial\Omega$ is chosen based on the priority values. Then, a target patch $\Psi_{\hat{p}}$ (represented in Figure 2b by the dashed square at the border of the target region $\partial\Omega$) is created surrounding the target pixel (represented by a large black dot in Figure 2b). From all the possible source patches within the search space $q \in \Phi$, the best match $\Psi_{\hat{q}}$ is determined (see Figure 2c). Next, as shown in Figure 2d, the pixels in the target area portion of the target patch $\Psi_{\hat{p}} \cap \Omega$ are filled with the corresponding pixels from the source patch $\Psi_{\hat{q}}$. The border of the target region $\partial\Omega$ is then updated and the confidence values of the newly copied pixels are set. These steps are repeated until the target area Ω is filled.

Here, to more clearly study the effect of focusing on image features rather than image regions, we have chosen to stick as closely as possible to Criminisi et al.'s algorithm in all stages with the sole exception of the size and shape of the source patch $\Psi_{\hat{q}}$. Of course, as mentioned before, the modular nature of this pipeline means that our proposed change can be combined with any of the other modifications. Thus, we start with a fixed-size square patch surrounding the target pixel. After using this target patch to find the correct match, the center of the source patch is used as a seed for a region-growing segmentation technique called a *morphological amoeba*, which captures the structure at the source region (see Section 2.3). We then copy only the pixels within the newly-grown source amoeba to the corresponding locations in the target area Ω. In the next few subsections, we provide more detail about the individual steps of the full algorithm.

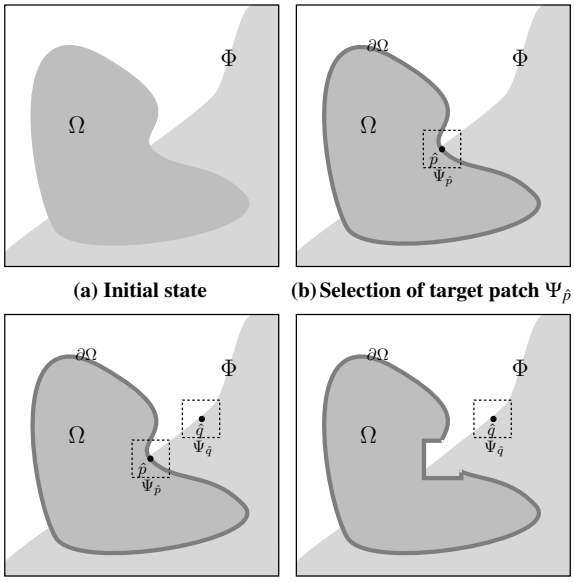
2.1 Target Patch Selection

Following Criminisi et al. [Cri04a], the first step in every iteration is the selection of the next target patch $\Psi_{\hat{p}}$ centering on the pixel with highest priority $\hat{p}$ at the contour $\partial\Omega$. All points $p \in \partial\Omega$ are sorted by a priority value $P(p)$, which takes into account two different factors, a data term and a confidence term:

$$P(p) = \varsigma(p) \cdot \gamma(p) \tag{1}$$

The confidence term $\gamma(p)$ is a measure of the reliability of the known image data around point p. It is the sum of the confidence value $C(i)$ of all the pixels i in the target patch divided by the number of pixels in the patch:

$$\gamma(p) = \frac{\sum_{i \in \Psi_p} C(i)}{\|\Psi_p\|} \tag{2}$$



(a) Initial state **(b) Selection of target patch $\Psi_{\hat{p}}$**

(c) Matching source patch $\Psi_{\hat{q}}$ **(d) Filling data in Ω**

Figure 2: Different steps of Criminisi's inpainting algorithm

Since all unknown pixels (the ones in the target area Ω) start with a confidence of 0 and all known pixels (the ones in the source area Φ) start with a confidence of 1, the more known pixels a patch has, the higher that patch's confidence value will be. After filling in, the newly filled-in pixels will receive a confidence less than 1 (see below for more details). As a result, patches close to the initial target-region border will have a higher confidence value than patches inside the (original) target area.

The data term $\varsigma(p)$ is a measure of the intensity of any linear structures pointing directly into the target area. A strong edge disappearing directly into the target area should be processed with higher priority than either a weak edge disappearing directly into the target area or a strong edge that is tangent to the target area. This will help to maintain the continuity of structural elements. The data term is defined as the normalized product of the dominant *isophote* $\nabla I^{\perp}$ in the target patch around a given point p and the normal vector $\vec{n}_p$ of the contour at that point:

$$\varsigma(p) = (\vec{n}_p \cdot \nabla I_p^{\perp})/\delta \tag{3}$$

with δ being a normalizing constant based on the possible pixel values and the isophote $\nabla I^{\perp}$ being defined as perpendicular to the intensity gradient. Thus, the direction of the isophote $\nabla I^{\perp}$ describes the orientation of the prevalent linear structure in the target area. To calculate the data term, we need to determine the dominant isophote in the target patch. To do this, we first calculate the isophotes for all pixels in the known portion of the target patch. Note that an isophote pointing directly upwards represents the same linear structure as

one pointing directly downwards, and as such we flip all isophotes with angles greater than 180 degrees so that they point in the opposite direction. We then examine the isophote histogram separately for each of the three color channels (with the angles now ranging between 0 and 179) and find the bin with the maximum summed gradient magnitude. These are the dominant isophotes of the patch, one for each color channel. The maximum of these three isophotes is chosen as the dominant isophote of the patch.

The pixel $p \in \partial\Omega$ with the highest priority value $P(p)$ is chosen as the center of the next target patch.

2.2 Source Patch Matching

To find the best match, we calculate the Euclidean color distance between each pixel in the known portion of the target patch $\Psi_{\hat{p}}$ and the corresponding pixels in all possible source patches. The source patch with the smallest summed color distance is chosen.

2.3 Amoeba

For this stage, which is novel to our algorithm, we start with the observation that copying a rectangular shaped region from the source area will have a non-zero probability of copying undesired structures (i.e, structures that are not the same as the one to which the target pixel belongs). Furthermore, once even a single pixel of an artifact has been copied to the target region, future in-painting iterations will consider the artifact to be a valid structure and will tend to complete the artifact. To avoid this, we propose that only the pixels in the source area that most closely resemble the target pixel itself (and thus are most likely to be on the same surface; see [Gi79a]) should be copied. We have chosen the morphological amoeba introduced in [Ler07a] to do this, as it utilizes both the physical distance between two pixels as well as the color distance. The amoeba determines which pixels belong to the patch by calculating the summed color distances and physical distances between pixels along a path. All pixels which can be connected to the target pixel by a path whose summed (color and spatial) distance is less than a given threshold are included in the patch. Conceptually, the amoeba starts by calculating the Euclidean color distance between all neighboring pixels p and q: $dist_{pixel}(p,q)$. It then calculates the summed color and physical distance $L(\sigma)$ between two pixels x and y along a given path σ:

$$L(\sigma) = \sum_{i=0}^{n-1} (PD + \lambda \cdot dist_{pixel}(x_i, x_{i+1})) \qquad (4)$$

where n is the number of pixels along the path and $\lambda \geq 0$ is a weighting factor which allows one to control the relative influence of the color distance over the physical distance. Note that the saliency of the physical distance between neighboring pixels in a Cartesian coordinate system is dependent on the viewing distance and the monitor resolution. Thus, the physical distance is the variable PD. Given the most common viewing distances and monitor sizes, the physical distance will typically be between 0.6 and 2. Typically, λ is set to 1, with the aim of ensuring that the physical distance and the color distance have equal saliency and equal importance. The final distance $d_\lambda(x,y)$ for two specific pixels is the path with the lowest $L(\sigma)$. Finally, all pixels y whose distance $d_\lambda(x,y)$ to the seed pixel x is below a user set threshold TH, belong to the amoeba.

2.4 Filling-in and Updating the Contour

In the next step, the data from the source patch is copied to the unknown portions of the target patch $\Psi_{\hat{p}} \cap \Omega$. Then, the confidence of the newly copied pixels is set to the average of all the pixels that are in an amoeba-shaped region centered on the target pixel. Finally, the target area Ω and the contour $\partial\Omega$ are updated and the priority values for all affected contour points are recalculated.

3 RESULTS

There are scenarios where image restoration methods will work best, such as images where the content to be generated lies in smooth or irregularly-textured areas, commonly present in natural images, or in areas that are not so likely to attract human attention. To gain more insight into effect of the new patch algorithm, we decided to use as a benchmark a set of considerably more challenging images. Specifically, we used part (see Figure 3) of the benchmark proposed by Rubinstein et al. [Rub10a]. This benchmark is known for containing images with attributes that represent a challenge for the objectives of preserving content and structure and preventing artifacts. Furthermore, we created the to-be-filled (target) areas in these images by selecting places that are most likely to be very challenging (e.g., that break local structures) and that are in areas most likely to attract human attention [Cas11a].

We submitted the 16 images to the original Criminisi et al. algorithm as well as to our modified version. We tested the effect of changing the size of the square target patch, using 20 different patch sizes. The twenty possible patch radii were in the range $r \in [1,20]$. Since a patch always included the target pixel and the number of pixels equal to the radius in each direction, this resulted in a range of patch sizes from 3×3 to 41×41 pixels. It is important to note that nearly all other tests of exemplar-based approaches use three patch sizes: 5×5, 7×7, 9×9 corresponding to the radii 2, 3 and 4. More rarely, 11×11 patches (radius of 5) are also used. We tested a *considerably larger range of patches* in order to more fully explore the range of image features that can be captured.

Figure 3: The 16 images used in our tests. The magenta-filled areas show the parts to be inpainted

For our algorithm, the maximum amoeba distance TH was set to 20 and the physical distance PD was set to 1.

The average run time of the amoeba-based algorithm is similar to the average run time of the unmodified original. It is important to note that the amoeba-based algorithm only needs to be run once to find a good patch size, whereas the Criminisi algorithm needs to be run once for each desired patch size.

4 VALIDATION

Ideally, in order to assess the quality of the results of both algorithms, a perceptual experiment with human participants should be conducted. Unfortunately, due to the large number of results to compared (with 20 patch sizes, 2 algorithms, and 16 images there are 640 resulting images), the duration of a perceptual experiment becomes untenable (e.g., a 2AFC preference task comparing each reconstruction for a given image to all the other reconstructions for the same image would require 12,480 trials per participant). Thus, we will use computational metrics to evaluate the image quality. Even if the metrics cannot replace the subjective evaluation of a human, they can be complementary and give us some insights for pre-filtering the results.

4.1 Metrics

The choice of metric is not a trivial issue. We can not use any metric which performs a pixel by pixel comparison since inpainting does not try to generate any specific texture, but instead focuses on perceptually plausible results. This means that a metric is needed that taps into the highly subjective issue of which image looks more plausible or natural. The choice of metric is further constrained by the central issue of patch size, as many existing image quality metrics break an image down into smaller patches and then analyze the image on a patch by patch basis. Unfortunately, these metrics all use a single, fixed patch size for any given image. It does not seem appropriate to use a fixed patch size to evaluate the effect of adaptive patch sizes. While it is appears interesting to construct a metric using adaptive patches, that is beyond the scope of this article.

Following [Rub10a, Cas11a], we selected two metrics that assess low-level differences between two images to give us an idea about the coherence of the image as a whole and how consistent the results are in comparison with the intact parts of the damaged image. These two metrics are Color Layout (CL) [Kat01a] and Edge Histogram (EH)[Man01a] (see below for more details).

We performed pairwise comparisons of the original image (without holes) and each of the inpainting results. The smaller the difference between the two images, the more closely the inpainting result matches the original.

These two metrics rely on statistical values that, even if they are good for measuring the amount of artifacts introduced, do not consider if these artifacts will be obvious to a human observer. Therefore, we also considered another metric, proposed by Ardis and colleagues [Ard10a] that relates the visual saliency map of an image with its perceived quality, the Average Squared Visual Salience (ASVS). For computing the metric's results we considered the bottom-up visual saliency model, Graph-Based Visual Saliency (GBVS) [Har06a].

Color Layout [*CL*]: The CL metric examines the differences in the distribution of color in YUV space between the images to compare:

$$CL = \sqrt{\sum_{i\in Y}\alpha_i(Y_i - Y_i')^2} + \sqrt{\sum_{i\in U}\beta_i(U_i - U_i')^2} + \sqrt{\sum_{i\in V}\gamma_i(V_i - V_i')^2} \quad (5)$$

where the ith coefficient of each channel is denoted by Y_i , U_i , V_i. The weights represented by α, β and γ are inversely proportional to the coefficient scan order.

Edge Histogram [*EH*]: The EH descriptor is capable of capturing the spatial distribution of edges in an image via a combination of 5-bin normalized histograms. To generate each histogram, the image is segmented in 4x4 pixel patches and the intensity component Y in the YUV color space is used to extract and classify the edges putting them into the 5-bins (vertical, horizontal, both diagonals and non-directional):

$$EH(I_O, I_R) = \|EH(I_O) - EH(I_R)\|_1 \quad (6)$$

Average Squared Visual Salience [*ASVS*]: ASVS is a non-reference metric that focuses on the impact that introduction of artifacts in the inpainted area causes in the viewer attention:

$$ASVS(I) = \frac{1}{|\Omega|}\sum_{\Omega}(S'_{I_R}(p))^2 \quad (7)$$

where $S'_{I_R}(p)$ is the saliency corresponding to a pixel in the target area of the inpainted result.

4.2 Results of the Metrics

The values given by the three image quality metrics for the two different algorithms, averaged over the 16 images and all patch sizes, can be seen in Table 1. Since all metrics give either a value for dissimilarity between two images or a value for impact of artifacts, a higher value represent worse results. As can be seen in the table, two of the three metrics agree that introducing the amoeba improved the image quality. Moreover, the two metrics that indicate better performance for the Amoeba algorithm are the metrics that more directly relate to our aim of removing visually disruptive intrusion artifacts in the inpainted results.

	Criminisi	**Amoeba**
ASVS	0.278	**0.260**
CL	**22.431**	23.448
EH	36.103	**30.776**

Table 1: Results for the Criminisi and Amoeba algorithms averaged over the 16 images and all radii. The metrics' results indicate that the Amoeba improves the quality of the results in several complementary aspects

The results are a bit more nuanced when one looks at the individual images and patch sizes (see Table 2). Here, we will examine the results for the Criminisi algorithm first, then the amoeba, and finally we will compare the two.

The Criminisi Algorithm Results: The first interesting result, as can be seen in the table, relates to the Criminisi algorithm. The radius that yielded the best result is rarely one the "standard" radii (2, 3, 4 or 5). Specifically, ASVS, EH, and CL metrics found the typical radii to be best just for 2, 3, and 4 of the 16 images, respectively. In other words, in 81% of the analyses, the best size was not in the typical range. Furthermore, for no image do all three metrics agree that the best radii is in the typical range and for only 1 image do two metrics agree the typical range is best. In short, the best radius for the Criminisi algorithm would never have been tested in previous work! Note that this explicitly means that the quality of the Criminisi reconstructions found in this paper will be of better quality than is typically expected from this algorithm.

The second interesting finding for the Criminisi algorithm is that there is no clear way of predicting which patch size is best. Changing the patch radius even by one often produces radically different image qualities. Likewise, patches with very different radii often have nearly identical scores. Critically, images with similar dimensions (like *tiger* and *twobirds*) show totally different trends for the optimal patch size, suggesting that different features are selected and the decision is not solely dependent on the image size. All these observations indicate that the only way of finding the best fixed patch size for the Criminisi algorithm is brute force, with the consequent exponential increase of computational time.

It is also interesting to note that, for Criminisi, the three metrics never agreed on what the best radius was. In fact, on only 5 of the 16 images did two metrics agreed. Clearly the metrics were focusing on different features.

	Size		Best radii for Criminisi			Best radii for Amoeba		
	W	H	ASVS	CL	EH	ASVS	CL	EH
bicycle	*460*	*300*	**10** (0.296)	**3** (15.71)	**9** (18.71)	**5** (0.285)	**4** (17.164)	**11** (14.623)
bungee	*206*	*308*	**4** (0.301)	**8** (29.72)	**5** (38.41)	**19** (0.256)	**20** (29.605)	**19** (35.064)
butterfly	*1024*	*700*	**20** (0.186)	**1** (17.63)	**20** (21.75)	**16** (0.172)	**16** (19.176)	**18** (16.654)
butterfly2	*615*	*422*	**1** (0.369)	**1** (11.66)	**20** (6.33)	**5** (0.439)	**7** (15.536)	**19** (7.271)
car	*500*	*375*	**17** (0.190)	**3** (8.06)	**3** (8.34)	**8** (0.191)	**2** (8.057)	**3** (10.439)
cat	*1024*	*683*	**18** (0.163)	**9** (36.24)	**15** (42.74)	**3** (0.191)	**1** (36.478)	**14** (35.484)
colosseum	*512*	*340*	**20** (0.027)	**2** (16.82)	**9** (34.37)	**14** (0.027)	**19** (16.662)	**19** (30.317)
eagle	*600*	*402*	**20** (0.142)	**2** (20.25)	**17** (27.26)	**7** (0.132)	**1** (22.165)	**20** (19.914)
glasses	*500*	*395*	**9** (0.233)	**6** (28.48)	**17** (80.02)	**2** (0.274)	**4** (35.712)	**3** (57.846)
mochizuki	*574*	*346*	**18** (0.270)	**1** (17.78)	**19** (19.89)	**9** (0.263)	**1** (19.714)	**17** (16.706)
mountains	*512*	*683*	**9** (0.440)	**17** (14.99)	**15** (8.32)	**17** (0.416)	**15** (14.489)	**4** (10.428)
penguins	*615*	*461*	**13** (0.217)	**15** (13.24)	**19** (18.96)	**19** (0.232)	**8** (18.316)	**11** (18.801)
pigeons	*800*	*600*	**7** (0.223)	**19** (11.07)	**5** (15.61)	**3** (0.218)	**12** (10.840)	**2** (18.696)
soccer	*500*	*356*	**13** (0.116)	**19** (32.96)	**7** (56.33)	**16** (0.085)	**4** (35.495)	**6** (49.288)
tiger	*600*	*437*	**12** (0.091)	**20** (13.70)	**20** (19.75)	**20** (0.099)	**5** (15.951)	**14** (19.743)
twobirds	*600*	*450*	**5** (0.434)	**1** (31.68)	**16** (39.77)	**12** (0.303)	**1** (36.107)	**12** (27.267)

Table 2: Best values according to the metrics. The bold numbers indicate the radii that produced the best results for the Criminisi and Amoeba algorithms. Numbers in parenthesis reflect the metrics' scores. As discussed in the text, it seems that our approach provides similar or better image quality in terms of edges and perceptual saliency

A closer examination of the different preferences provides some useful insights. The CL metric prefers small patches (for 8 of the 16 images, the best size was a radius of 3 or smaller). For 5 images, CL preferred a large patch (15 or higher). A closer examination of the images suggests that this difference is being driven by the image features. The large patch size is preferred when the target region is inside a mostly homogeneous region (such as for *tiger* or *mountains*) and small when the target region has lots of texture or edges (such as *eagle* or *colosseum*). The EH metric, on the other hand, strongly prefers large patches (for 10 images the best radius was 15 or higher), and only once prefers small patches. This sole case where EH preferred a small patch was for an image (*car*) that had target regions with a few dominant edges at different spatial scales. The ASVS, which focused on visual saliency, preferred medium (9 images) or large (6 images) patches, and only once preferred small images.

The Amoeba Algorithm Results: As with Criminisi, the standard sized patches were not chosen very often: in 73% of the analyses, the best size was not in the typical range! Likewise, the three metrics never agreed on what the best patch size was. For only four of the images did two of the three metrics agree on the best patch size.

The preferences of the individual metrics was also similar to that of the Criminisi Algorithm. Specifically, the CL metric seems to prefer small patch sizes (for 5 of images the patch size was 3 or smaller, and for 8 o the images it was 4 or smaller). The EH metric seems to prefer large patches (for 6 images the best radius was 15 or higher; for 8 images it was 14 or higher). Smaller patches did, however, performed better than in the Criminisi algorithm. The ASVS preferred middle and large patches (6 at 15 or more; and only 3 with a radius of 3 or less).

Comparing Algorithms: Despite the same general trends for the preferences of the three metrics, the specific patch size that was seen as best in the two algorithm was very different. Only in three of the 48 analyses (16 images for 3 metrics) did a metric favor the same patch size for the same image in both algorithms. It is clear that the amoeba drastically altered the image reconstruction. As would be expected from the above discussion, there are few cases where either algorithm is a clear winner in terms of the computational metrics, as documented in Table 2. For 9 of the 16 images the ASVS felt that the amoeba outperformed the Criminisi algorithm and in 2 occasions the scores were tied. The EH metric felt that the amoeba outperformed the Criminisi algorithm in 12 of the 16 images. The CL metric, on the other hand, felt that the amoeba outperformed the Criminisi algorithm on a mere 5 images.

4.3 Face Validity

Given that the metrics rarely agree with each other, it is clear that we cannot rely on them too much without additional information about their relationship to perception. The relationship between the metrics and perception can be seen a bit more clearly in Figure 4 which shows the best image for both algorithms for each algorithm. For the image *tiger*, for example, two of the metrics rated Criminisi as better. A closer look at the images, however, shows that for each metric, our version had fewer intrusion artifacts and abrupt discontinuities. The image chosen by EH for our algorithm is clearly the most natural reconstruction. For the image *soccer*, only

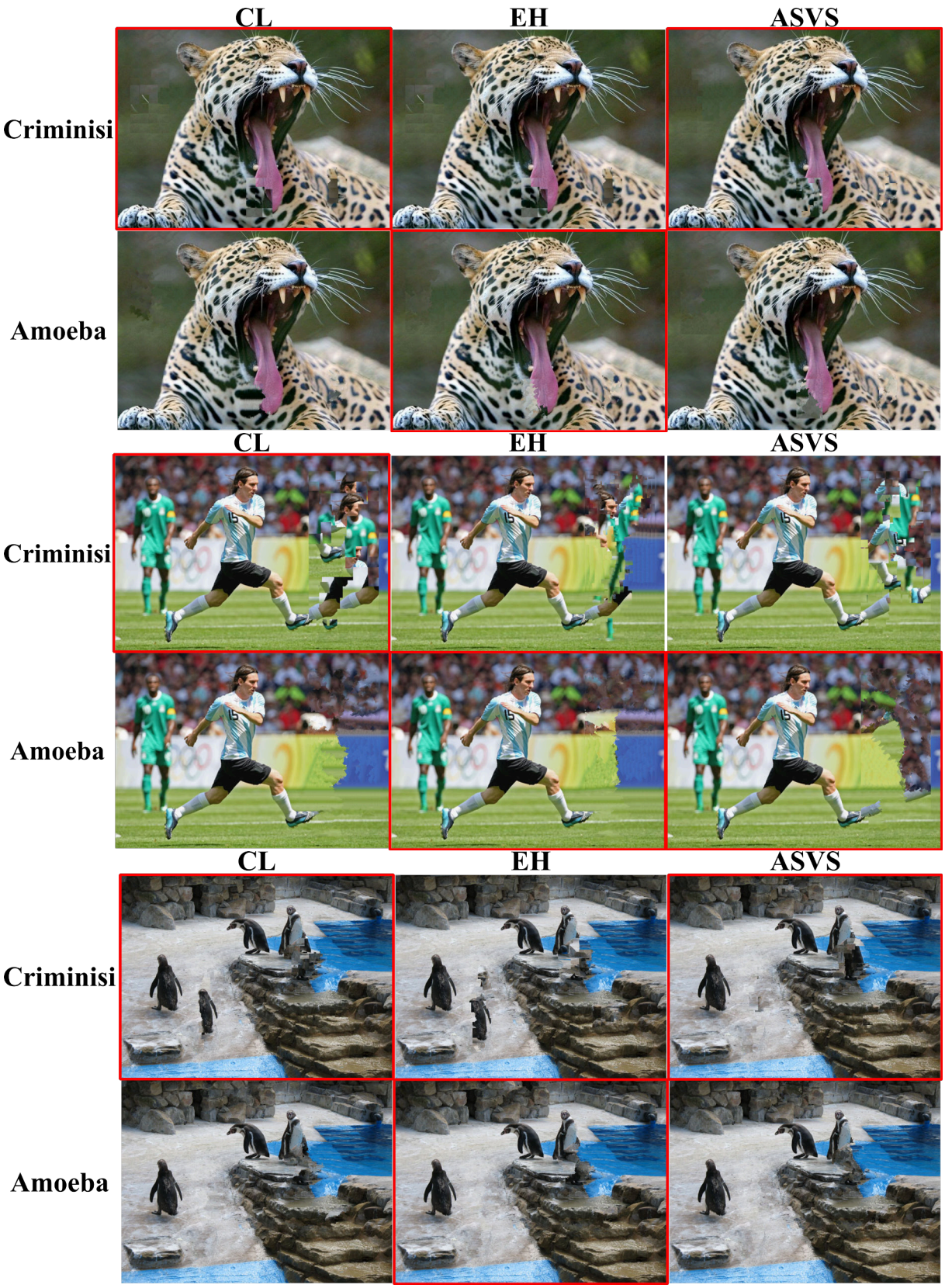


Figure 4: The best reconstruction according to the three metrics for the two algorithms. Each metric's preferred algorithm is highlighted with a red frame. As seen in the images the Amoeba algorithm results have visually fewer intrusions. For reference, the first row shows the original images without holes

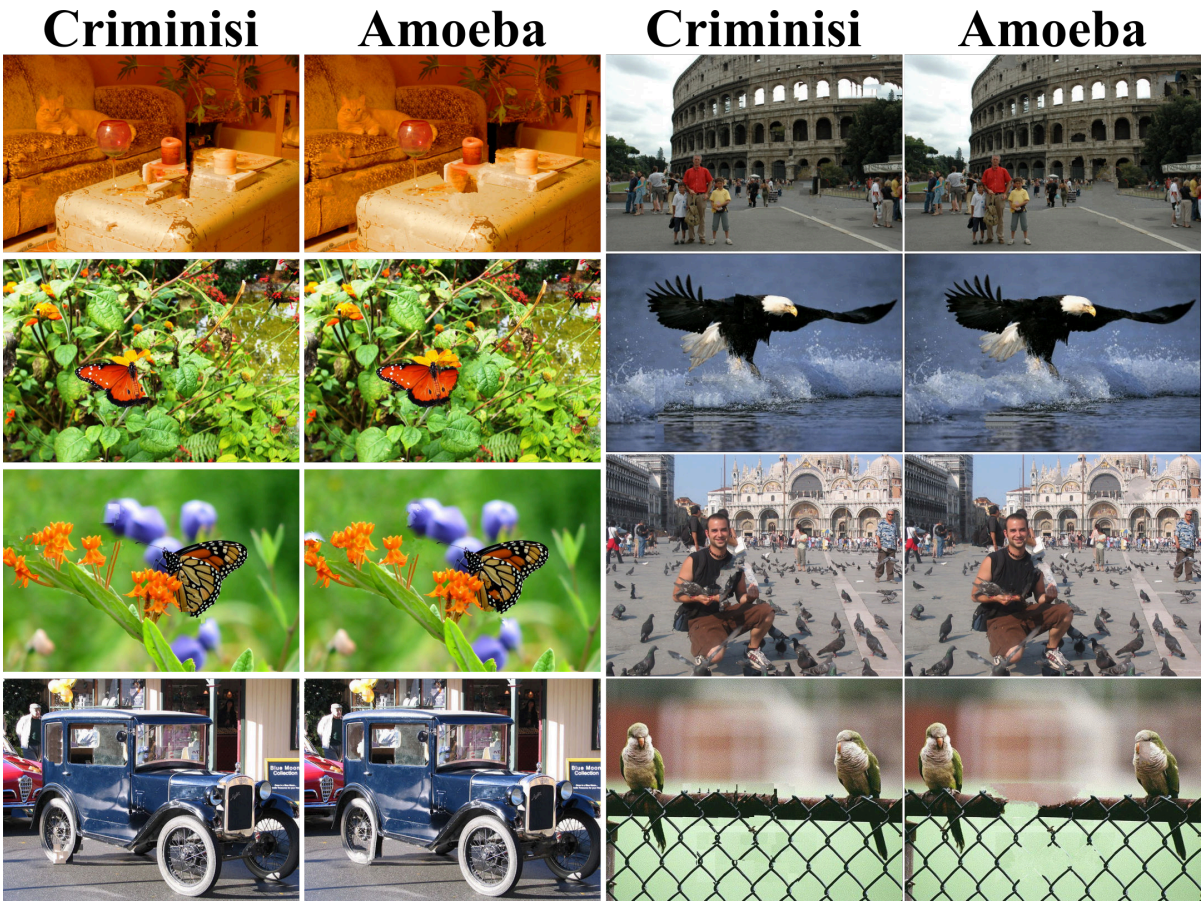


Figure 5: Subjectively selected best reconstructions for the two algorithms

CL preferred Criminisi. Yet clearly all of the chosen images for our algorithm contain fewer intrusion artifacts, and the results are definitely more perceptually plausible. It seems that the EH and to some degree the ASVS metrics are better at predicting the level of visual salience of disruptive intrusion artifacts. Finally, the image *penguins* two metrics preferred the results from Criminisi algorithm. Here again, we would argue that a closer look at the image reveals disconcerting artifacts in our reconstructions.

Since clearly the metrics are not very good at predicting human performance, with the possible exception of EH, it is possible that the best images chosen by the metrics, as seen in Figure 4, might not the best set of image to compare the two pipelines. Therefore, we examined the entire set of results (all 640 image) and presented them to several observers to choose the subjectively best image. A representative sample of the perceptually best image can be seen in Figure 5. Although clearly both algorithms can at times produce good reconstructions, it seems clear that the amoeba version has fewer intrusion artifacts and fewer disruptive or abrupt completions.

5 CONCLUSIONS

We extended the method and analysis of Criminisi et al. [Cri04a] in two ways. First, we showed that the best sized rectangular patches for the original, unmodified Criminisi algorithm are almost always well outside the range usually tested. Second, we demonstrated that growing a non-rectangular patch in the source area allows the algorithm to copy only pixels that most closely resemble the structure near the target pixel greatly reduces the chance of inserting artifacts into the target area and consequently improves image quality. This can be seen clearly in the figures and was confirmed by the image quality statistics. The amoeba we proposed has considerably reduced the occurrence of disembodied, partially completed, oddly placed structures. Given that this modification alters the size and shape of the patch itself, its effects are orthogonal to nearly all the other modifications to the method of Criminisi et al. made by other researchers and as such can be combined with them. Of course, matching and copying techniques that explicitly require a square patch will require modification to work with the new amoeba patches. Future work could focus on which of the many modules for each of the different stages in Criminisi's pipeline works optimally. Future work could also examine using the amoeba for the target patch as well. Finally, it is likely that amoebae can conceivably be used in any technique that employs patches to process, synthesize, or analyze images.

6 ACKNOWLEDGEMENTS

The authors would like to thank Stephan Guthe and Maximilian Mühle for their suggestions and valuable comments.

7 REFERENCES

[Ad17a] R.D. Adam, P. Peter, and J. Weickert. Denosing by inpainting. In *Proc. SSVM '17*, 121–132, 2017.

[Ard10a] P.A. Ardis, C.M. Brown, and A. Singhal. Inpainting quality assessment. *Journal of Electronic Imaging*, 19(1):011002–011002–7, 2010.

[Ber03a] M. Bertalmió, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Trans. Image Process.*, 12(8):882–889, 2003.

[Ber00a] M. Bertalmió, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. SIGGRAPH '00*, 417–424, 2000.

[Buy15a] P. Buyssens, M. Daisy, D. Tschumperlé, and O. Lézoray. Exemplar-based inpainting: Technical review and new heuristics for better geometric reconstructions. *IEEE Trans. Image Process.*, 24(6):1809–1824, 2015.

[Cai17a] N. Cai, Z. Su, Z. Lin, H. Wang, Z. Yang, and B.W.-K. Ling. Blind inpainting using the fully convolutional neural network. *The Visual Computer*, 33(2):249–261, 2017.

[Cas11a] S. Castillo, T. Judd, and D. Gutierrez. Using eye-tracking to assess different image retargeting methods. In *Proc. APGV '11*, 7–14, 2011.

[Cri04a] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. Image Process.*, 13(9):1200–1212, 2004.

[Ef99a] A.A. Efros and T.K. Leung. Texture synthesis by non-parametric sampling. In *Proc. ICCV*, vol. 2, 1033–1038, 1999.

[Gi79a] James J. Gibson. *The ecological approach to visual perception*. Boston: Houghton Mifflin, 1979.

[Gui14a] C. Guillemot and O. Le Meur. Image inpainting : Overview and recent advances. *IEEE Signal Process. Mag.*, 31(1):127–144, 2014.

[Har06a] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *Proc. NIPS*, pp. 545–552, 2006.

[Kat01a] E. Kasutani and A. Yamada. The mpeg-7 color layout descriptor: a compact image feature description for high-speed image/video segment retrieval. In *Proc. ICIP '01*, vol. 1, 674–677, 2001.

[Kom07a] N. Komodakis and G. Tziritas. Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *IEEE Trans. Image Process.*, 16(11):2649–2661, 2007.

[Lem13a] O. Le Meur, M. Ebdelli, and C. Guillemot. Hierarchical super-resolution-based inpainting. *IEEE Trans. Image Process.*, 22(10):3779–3790, 2013.

[Lem11a] O. Le Meur, J. Gautier, and C. Guillemot. Examplar-based inpainting based on local geometry. In *Proc. ICIP*, 3401–3404, 2011.

[Lee12a] J. Lee, D. K. Lee, and R. H. Park. Robust exemplar-based inpainting algorithm using region segmentation. *IEEE Trans. Consum. Electron.*, 58(2):553–561, 2012.

[Ler07a] R. Lerallut, É. Decencière, and F. Meyer. Image filtering using morphological amoebas. *Image and Vision Computing*, 25(4):395–404, 2007.

[Liu08a] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W.T. Freeman. Sift flow: Dense correspondence across different scenes. In *Proc. ECCV*, 28–42, 2008.

[Mai09] M. Mainberger and J. Weickert. Edge-based image compression with homogeneous diffusion. In *Proc. CAIP*, 476–483, 2009.

[Man01a] B. S. Manjunath, J. R. Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *IEEE Trans. Circuits Syst. Video Technol.*, 11(6):703–715, 2001.

[Mas98a] S. Masnou and J.M. Morel. Level lines based disocclusion. In *Proc. ICIP*, vol. 3, 259–263, 1998.

[Ngu13a] H. M. Nguyen, B. C. Wünsche, P. Delmas and C. Lutteroth, Parameter optimisation for texture completion. In *Proc. IVCNZ '13*, 226–230, 2013.

[Rub10a] M. Rubinstein, D. Gutierrez, O. Sorkine, and A. Shamir. A comparative study of image retargeting. *ACM Trans. Graph.*, 29(6):160:1–160:10, 2010.

[Wu09a] J.Y. Wu and Q.Q. Ruan. A novel exemplar-based image completion model. *J. of Information Science and Engineering*, 25:481–497, 2009.

[Xi13a] X. Xi, F. Wang, and Y. Liu. Improved Criminisi algorithm based on a new priority function with the gray entropy. *Proc. CIS*, 214–218, 2013.

[Xu10a] Z. Xu and J. Sun. Image inpainting by patch propagation using patch sparsity. *IEEE Trans. Image Process.*, 19(5):1153–1165, 2010.