Parameter Optimisation for Texture Completion

Hoang M. Nguyen, Burkhard C. Wünsche, Patrice Delmas, and Christof Lutteroth
Dept of Computer Science, University of Auckland, Auckland, New Zealand
justin.nguyen@auckland.ac.nz {burkhard, p.delmas, lutteroth}@cs.auckland.ac.nz

Abstract—In 3D reconstruction applications frequently large mesh sections are missing. While the geometry can be created with mesh completion techniques, reconstructing the texture is more difficult. An exemplar-based synthesis usually requires that the missing region contains only one texture pattern. Image inpainting techniques work for more complex textures, but were designed primarily to fill narrow missing regions and tend to produce undesirable results when the missing region is large. We solve this problem by exploiting the fact that most 3D objects contain different surface regions with similar textures. We use an appearance space to identify suitable texture patches fitting with the boundary of the missing region, and fill the region by using a patch-based synthesis combined with Poisson interpolation. We evaluate the effect of different parameters and demonstrate its improved performance over existing inpainting techniques.

Keywords—texture retouching, texture inpainting, texture completion, image information recovery

1 Introduction

Over the past couple of years 3D reconstruction technologies have improved tremendously and are now available in the consumer-level domain. Examples include structured lighting (Kinect), laser scanners, and image-based modeling. In many applications the resulting mesh contains large missing regions, e.g. because the surface was invisible to the utilised sensor. Surface reconstruction techniques, such as Poisson Surface Reconstruction, and mesh completion techniques can create a watertight surface, but are unable to reconstruct texture information.

Missing textures can be obtained with texture synthesis methods, but these usually employ a single exemplar and assume a consistent texture over the missing region. Image inpainting techniques work for more complex textures, but were designed primarily to fill narrow missing regions and tend to produce undesirable results when the missing region is large.

3D objects in the real-world usually contain repeating geometries and textures, e.g. the sides of an animal, components of a plant (stem, branch, blossom), or man-made objects which are usually designed using symmetries and self-similarities (e.g. walls and windows of a house).

In this paper we perform texture completion by exploiting the fact that most 3D objects contain different surface regions with similar textures. We use an appearance space to identify suitable texture patches fitting with the boundary of the missing regions, and fill the region by using a patch-based synthesis combined with Poisson interpolation.

The remainder of this paper is organised as follows: After a brief discussion of existing image inpainting techniques in section 2, we describe our inpainting algorithm in section 3. Section 4 presents results and section 5 concludes this paper.

2 Related Work

Image in-painting techniques can be divided into two classes. Pixel-based methods fill the missing region pixel-by-pixel starting from the boundary, whereas patch-based methods usually add entire patches and minimise discontinuities between texture regions.

The arguably best known and most successful algorithm amongst pixel-based inpainting methods was proposed by Bertalmio *et al.* [1]. The authors attempt to replicate manual inpainting by propagating the known color values into a missing region along so called isophotes, representing the smallest spatial change of color values and structures.

Drori *et al.* [2] use adaptive circular fragments to operate on different scales to capture both global and local structures and approximate the missing region.

Telea *et al.* [6] present an inpainting technique based on a fast marching method for level set applications. The method is simple and considerably more efficient than other pixel-based methods.

Pixel-based methods often fail to properly reconstruct larger structures with semantic meaning, e.g. leaves. Criminisi *et al.* [3] iteratively select a "best-fit" rectangular patch and copy it over to the target region. The order in which boundary pixels of the missing region are processed is based on the amount of information available for that pixel and whether it has any prominent features.

Cheng *et al.* [4] update the priority equation of [3] and made it adjustable to the structural and textural information specific to an image. Ignacio *et al.* [5] extend the concept of Criminisi's method and apply it in the wavelet domain.

3 Algorithm

We employ a similar patch search and insertion concept as Criminisi *et al.*, but use appearance space attributes and principal component analysis to improve region matching quality and speed. In addition, our method smoothly fuses patches together to remove all visible seams.

3.1 Candidate Patch Identification

The algorithm's performance is significantly effected by the ability to identify the patch in the image that retains the highest resemblance to the processed patch. This is achieved by iteratively traversing through each pixel of the image outside the missing region and computing the similarity of the patch centered around that pixel and the original patch. Instead of using the standard *Sum of Squared Differences (SSD)* to measure the similarity of two given patches, we employ *appearance space attributes*, which provide much more information and thus improve the search result.

When searching for a matching patch, we consider for each pixel an 11×11 pixel neighbourhood. For each pixel of this neighborhood we consider *RGB* colours, the *gradient vector*, as well as the signed Euclidean distance to the closest dominant feature in the original texture.

Gradient Vector: We estimate the gradient of an intensity image $\mathbf{I}$, $(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$ by using the convolution kernels of a standard Sobel operator. The two kernels for the x and y directions are:

$$S_x = (1\ 2\ 1)^T \cdot (1\ 0\ -1) \tag{1}$$

$$S_y = (1\ 0\ -1)^T \cdot (1\ 2\ 1) \tag{2}$$

The attribute gradient image is then defined as:

$$\mathbf{A}(p) = ((S_x * \mathbf{I})(p), (S_y * \mathbf{I})(p)) \tag{3}$$

where p denotes an image pixel.

Signed Feature Distance: The signed feature distance of a pixel p is defined as the distance of p to the closest pixel $q \in \mathbf{M}$ for which $\mathbf{M}(p) \neq \mathbf{M}(q)$ where $\mathbf{M}$ is a binary image created by applying the *canny edge detection* method [11] on the input image.

The distance between two pixels is defined using the following equations (adapted from [9]):

$$\mathbf{A}(p) = s \cdot (\underset{r \in \mathbf{M}}{max}\ \lambda(r) - \lambda(p)) \tag{4}$$

$$\lambda(x) = \frac{1}{|q_x - x|} \tag{5}$$

$$f(x) = \begin{cases} 1 & if\ \mathbf{M}(p) = 1 \\ -1 & if\ \mathbf{M}(p) = 0 \end{cases} \tag{6}$$

The vector $(q_x - x)$ points from x to the closest pixel q in $\mathbf{M}$.

The entire attribute information is encapsulated into a $11 \times 11 \times (3 + 2 + 1) = 726$-dimensional vector. Determining the similarity of two given patches by comparing two 726-dimensional vectors is not efficient. In order to make the appearance space more practicable, the 726 dimensional vectors are projected into low-dimensional vectors using *principal component analysis (PCA)* ([9], [10]). In our method, the dimensionality is reduced to 8, which in our experiments with different image types produced the best results.

The clear advantage of the attribute space over the conventional SSD is that the attribute space approach permits any meaningful information about the pixels and their surrounding to be embedded for matching purposes. By reducing the dimensionality, the computation time can be kept manageable.

3.2 Patch Fusion

The final step is to replicate the content of the candidate patch and smoothly blend it with the target region. We employ a *Poisson-guided interpolation* approach proposed by [7] for this task.

The goal is to adjust the colour information of patch Ψ_B, while preserving the *relative information* (image gradient) as much as possible, so that the transition between the newly modified patch Ψ_C and the rest of the image is gracefully blended.

4 EVALUATION

In this section, we investigate the effect of different algorithm parameters and evaluate its performance over popular existing algorithms.

4.1 Effect of Parameters

Fig. 1. The original image is on the left, while the damaged image is on the right

Effect of appearance space attributes: Figure 2 shows an example in which the previous damaged "lizard" image (Figure 1) is inpainted using various different appearance space attributes. The following combinations of appearance space attributes are used for testing.

- Case 1: RGB color, HSB color, signed feature distance, horizontal and vertical gradient vectors.

a

b

c

d

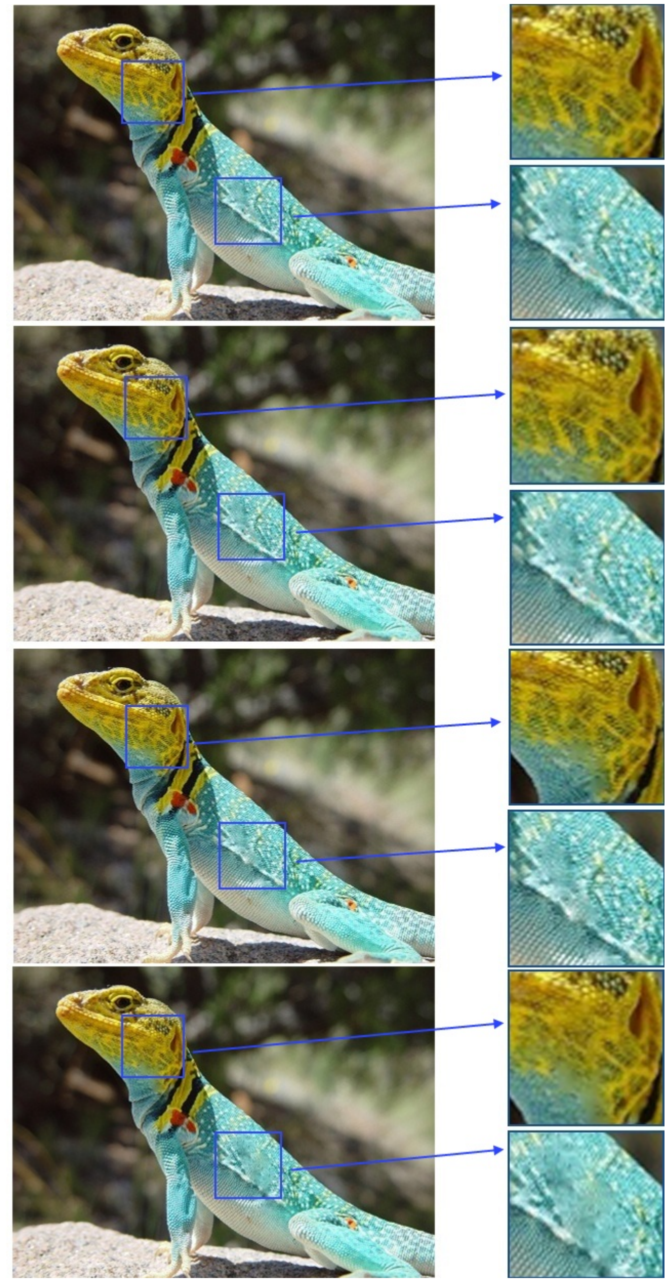

Fig. 2. Appearance space attribute parameters: a) Color, HSB color, signed feature distance, horizontal and vertical gradient vectors. b) Color, signed feature distance, horizontal and vertical gradient vectors. c) Color and signed feature distance. d) Color, horizontal and vertical gradient vectors.

- Case 2: RGB color, signed feature distance, horizontal and vertical gradient vectors.
- Case 3: RGB color and signed feature distance.
- Case 4: RGB color, horizontal and vertical

gradient vectors.

As can be observed from figure 2, there is little or no visual difference between the first and second case. This indicates that adding the HSB color attribute does not result in finding better patches.

There is an obvious difference in the result quality when the gradient vectors are removed. Some parts of the texture of the head become fuzzy and appear incorrect. A small part of the line on the body of the lizard is missing.

The result quality further deteriorates when removing the signed feature distance attribute in favor of the gradient vectors. The textures become much more blurry and more missing features are noted. This is probably due to the fact that inccorect patches have been identified and blended in this case.

We tested the different combinations of attributes with various images from different domains, and the results indicate that the optimal combination of appearance space attributes is *RGB color*, *signed feature distance*, *horizontal and vertical gradient vectors*. Adding more attributes often contributes little or no improvement and yet increases the computational cost.

Effect of attribute vectors' dimensionality: In order to improve the efficiency, appearance attribute vectors are projected to a low-dimensional vectors before being compared to others. In this section, we evaluate the effects of the dimensionality of reduced appearance attribute vectors on the the result quality.

Figure 3 shows an example in which appearance attribute vectors were reduced to different dimensions (8, 7, 6, 5) before comparison. Reducing the original $726-$dimensional attribute vectors to a $8-$dimensional vector yields good results. Missing textures are reconstructed well and seamlessly fused with the existing textures. We found that using more than 8 dimensions does not result in a visible improvement of the texture completion result.

Reducing the dimension further to 6 or 7 increases blurriness. This is probably due to the fact that the reduced vectors did not contain enough information for correct patches to be found.

When only 5 dimensions are retained, deformations start to appear due to the lack of information required for correct patch detections.

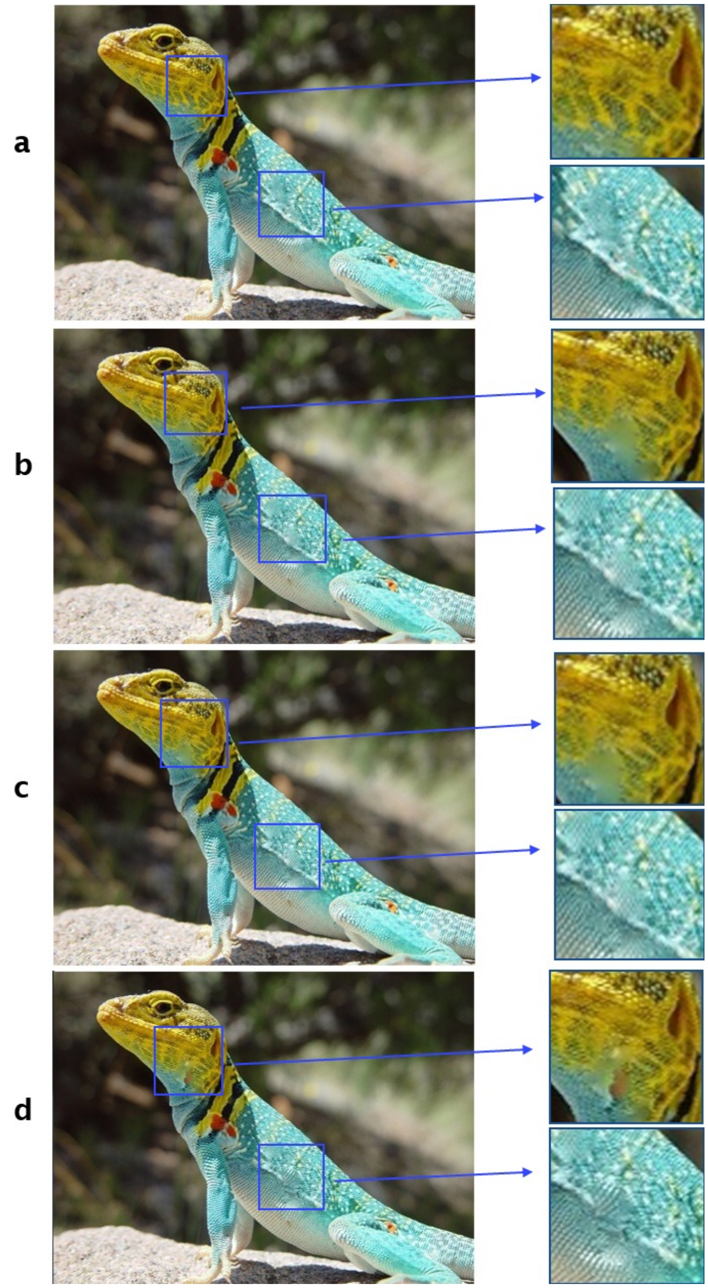

Fig. 3. Appearance space attribute dimensions: a) $8-$dimensional vectors. b) $7-$dimensional vectors. c) $6-$dimensional vectors. d) $5-$dimensional vectors.

Effect of Patch Size: Figure 4 demonstrates the difference in the result quality when varying the patch sizes. We found that in most cases the ideal patch size is the range between 7 and 11. Inpainted images in these cases are often well-reconstructed. As the size of the patches increases, the painted regions tend to become more blurry. This is probably because the larger the patch size is, the more unrelated features from surrounding regions are involuntarily taken into account leading to less accurate patch to be selected. Patch sizes smaller than 7 increase the computational cost while contribute little improvement in term of the quality.

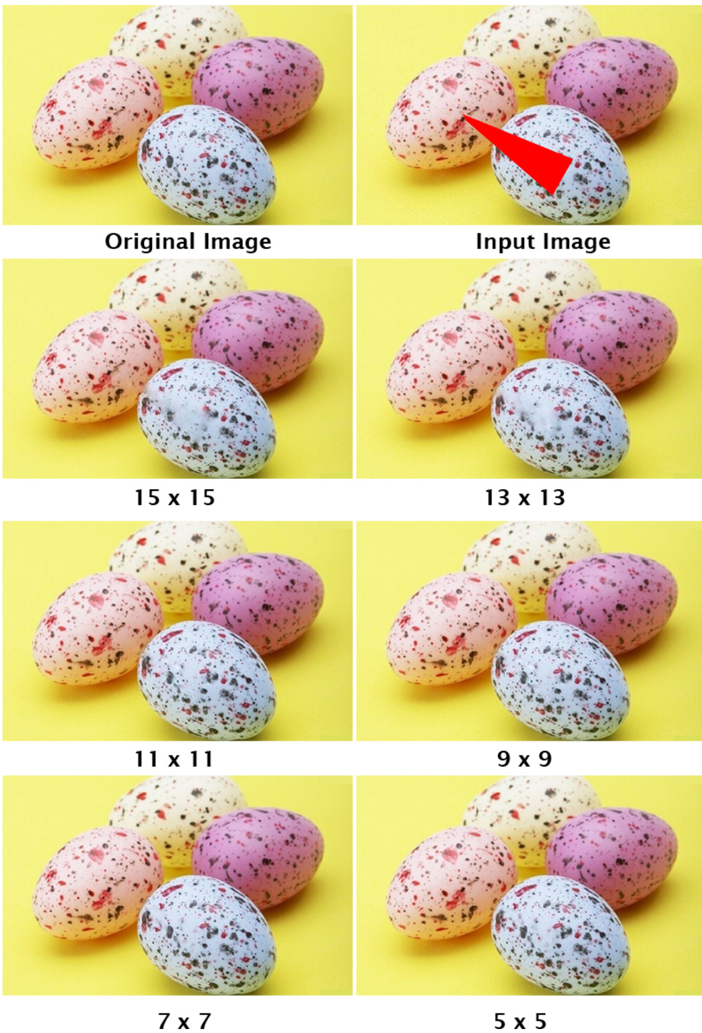

Fig. 4. Effect of patch sizes on the result quality.

4.2 Evaluation against Other Inpainting Methods

In this section, we evaluate the performance of our method against some of the best known image inpainting methods described in the literature (Figure 5).

Bertalmio's method [1] was not very successful in this test. It was able to reconstruct small parts of the missing regions (near the boundary), but failed to interpolate textures further for the middle regions resulting in patches with colours not consistent with the regions neighborhood.

Telea's method [6] outperformed Bertalmio's method both in terms of reconstruction quality and efficiency. However, the inpainted regions appear very blurry and unrealistic. This is expected as pixel-based methods were designed to tackle only narrow missing regions.

Standard exemplar-based inpainting method (Criminisi [3]) performed reasonably well except for some artefacts. These artefacts are caused by incorrect patches that were selected as a result of employing the conventional *SSD*.

Our algorithm performed well in this test case. Although the inpainted region still exhibits slight blurriness, the overall structure of the different scene components is correctly recovered.

5 Conclusion and Future Work

We have presented a novel image inpainting algorithm for synthesizing large missing texture regions from digital images. The results of this inpainting process is a new image in which the deterioration has been "inpainted" and reverted in such a way that few visible traces of it remain. The basic idea of our approach is to replicate missing textures by looking for "best-fit" texture patches in the source regions and smoothly inserting these patches into the missing region to produce the final result.

Our solution offers two major improvements compared to existing techniques. Patches for filling in missing regions are found using an appearance space vector, which not only encodes colour differences between regions, but also colour gradients, feature distances and other measures for image similarity. The second major improvement is the technique to combine the patches filling in a missing region. We use a Poisson-guided interpolation to blend patches to avoid visible seams.

We have evaluated our method's performance against some of the best known inpainting methods described in the literature and found that our results are superior.

References

[1] Marcelo Bertalmio and Guillermo Sapiro and Vicent Caselles and Coloma Ballester, Image Inpainting, In Proceeding SIGGRAPH '00 Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 417–424

[2] Iddo Drori and Daniel Cohen-Or and Hezy Yeshurun, Fragment-Based Image Completion, ACM Transactions on Graphics, Vol. 22, pp. 303–312

[3] A. Criminisi and P. Perez and K. Toyama, Object removal by exemplar-based inpainting, ACM Transactions on Graphics, pp. 721–728

[4] Wen-Huang Cheng and Chun-Wei Hsieh and Sheng-Kai Lin and Chia-Wei Wang and Ja-Ling Wu, Robust algorithm for exemplar-based image inpainting, In Proceedings of CGIV, pp. 64–69

[5] Ubirate Ignecio and Cleudio R Jung, Block-based image inpainting in the wavelet domain, Visual Computing, pp. 733–741

[6] Alexandru Telea, An image inpainting technique based on the fast marching method, Journal of Graphics Tool, Vol. 9, pp. 23–34

[7] Patrick Perez and Michel Gangnet and Andrew Blake, Poisson image editing, ACM Transactions on Graphics, Vol. 22(3), pp. 313–318

[8] Alexei Efros and Thomas Leung, Texture synthesis by non-parametric sampling, In Proceeding of ICCV, pp. 1033–1038

[9] Felix Manke and Burkhard Wunsche, Analysis of appearance space attributes for texture synthesis and morphing, Image and Vision Computing New Zealand 2009. IVCNZ 09, pp. 85–90

[10] Sylvain Lefebvre and Hugues Hoppe, Appearance-space texture synthesis, ACM SIGGRAPH 2006 Papers, pp. 541–548

[11] Canny J, A Computational Approach To Edge Detection, IEEE Trans. Pattern Analysis and Machine Intelligence 1986, Vol. 8(6), pp. 679698

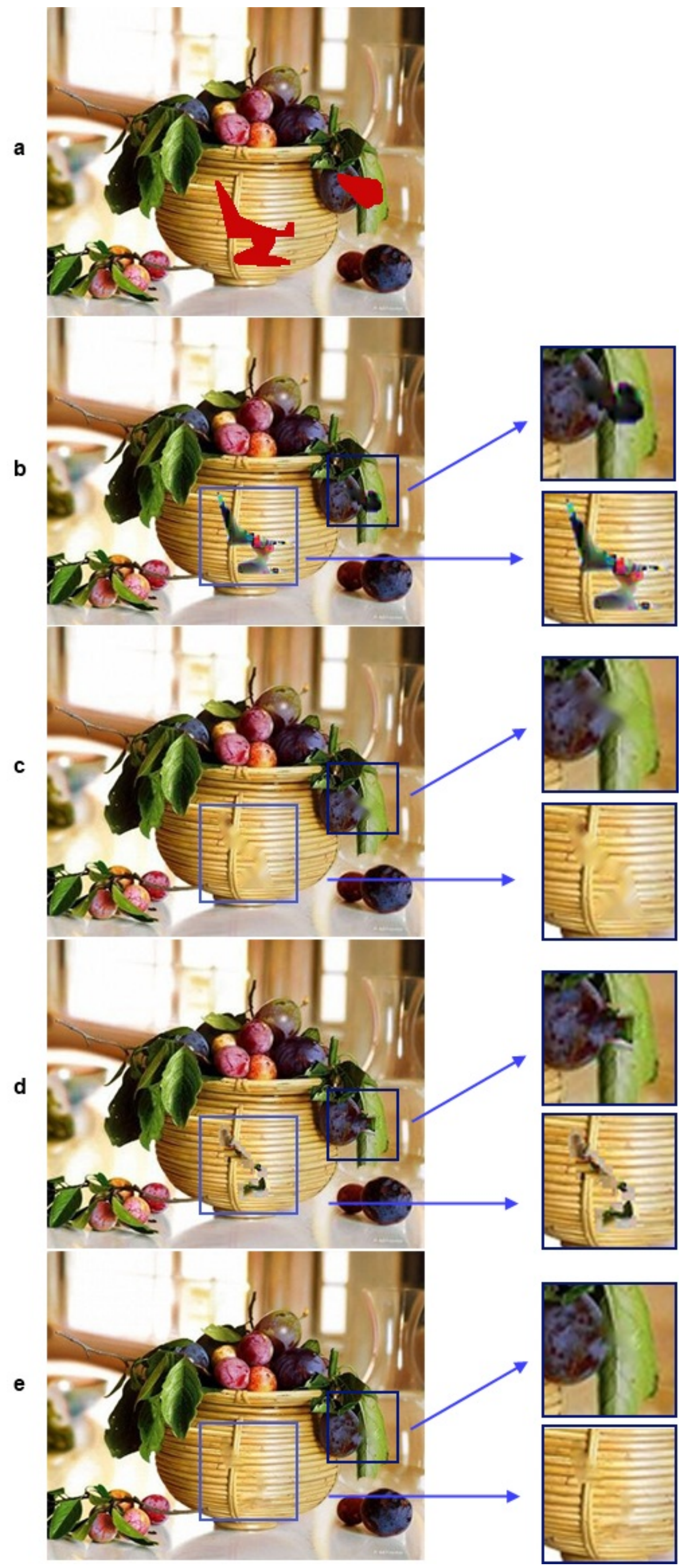

Fig. 5. a) The input image. Image inpainting results obtained using the algorithms from: (b) (Bertalmio et al. 2000), (c) (Telea et al. 2004), (d) (Criminisi et al. 2003) and (e) our method.