Novel Techniques for Auto-inpainting in Heritage Reconstruction

by

PADALKAR MILIND GAJANAN
201121015

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY
in
INFORMATION AND COMMUNICATION TECHNOLOGY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

September, 2016

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Doctor of Philosophy in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

Padalkar Milind Gajanan

Certificate

This is to certify that the thesis work entitled NOVEL TECHNIQUES FOR AUTO-INPAINTING IN HERITAGE RECONSTRUCTION has been carried out by PADALKAR MILIND GAJANAN for the degree of Doctor of Philosophy in Information and Communication Technology at *Dhirubhai Ambani Institute of Information and Communication Technology* under my supervision.

Prof. Manjunath V. Joshi
Thesis Supervisor

Acknowledgments

First and above all, I praise God, the almighty for providing me this opportunity and granting me the capability to proceed successfully. Although the work described here was performed independently, I never would have been able to complete it if not for the support of many wonderful people. I would like to offer my sincere thanks to them.

I would like begin by expressing my sincere gratitude to my supervisor Prof. Manjunath V. Joshi, with whom I have learned immensely and has had a strong influence in my development as a researcher. He has constantly encouraged me to ensure that I remain focused on achieving my goal. I am grateful to him for patiently supervising and directing my work, fruitful discussions, providing learning opportunities on a number of occasions, and helping me throughout all the different steps of my doctoral research endeavor for the past few years. I would also like to thank him for giving me an opportunity to work in a project sponsored by the Dept. of Science and Technology, Govt. of India, which not only provided me great learning experiences but also funding to partially support my doctoral studies. Prof. Joshi's achievements, his work ethics, and his keen eye for every important detail have been an inspiration throughout all the years I have worked with him.

On a broader note, I wish to acknowledge all the professors of DA-IICT who have inspired me directly or indirectly. I would like to thank faculty members Prof. Suman Mitra, Prof. Asim Banerjee, Prof. Hemant Patil and Prof. Manish Narwaria who have shaped my thinking about this research direction during the thesis work. I express my gratitude to Prof. N. Ramrao (Director), Mr. Soman Nair (Registrar), Prof. Anish Mathurai (former Dean-R&D), Prof. Laxminarayana

Pillutla (Convener-ICT), Prof. Aditya Tatu (former Ph.D. Coordinator) and Prof. V. Sunitha (M.Tech. Coordinator) who have helped me throughout my time at DA-IICT. I would like to thank the placement officer Mr. Sunil Jain, administrative and technical staff members, Mrs. Anuradha Srivastav, Mr. Mukesh Thaker, Mr. Ramesh Prajapati, Mr. Rajendra Shah and help-desk of DA-IICT who have been kind enough to advise and help me in their respective roles. My gratitude also extends to staff of DA-IICT resource-center.

I would like to express my gratitude to Prof. Mukesh Zaveri (Sardar Vallabhbhai National Institute of Technology (SVNIT), Surat) for urging me to pursue doctoral research and constantly encouraging me in this endeavor. I would also like to thank Prof. Mehul Raval (Ahmedabad University) for his valuable inputs. I thank Prof. Toshiyuki Amano, (Wakayama University, Japan) for his valuable communication over emails and generosity in sharing the code of his work in [4]. I express my sincere thanks to my teachers Prof. Santosh V. Jadhav (Finolex Academy of Management and Technology (FAMT), Ratnagiri) and Mr. P. A. Thomas (Vidya Vikasini English High School, Vasai) who have constantly motivated me to work with complete dedication and determination in a disciplined manner. I am grateful to Xerox Research Centre India for supporting me with a travel grant to present our work at British Machine Vision Conference 2015 (BMVC-2015). I would also like to thank all the anonymous reviewers of our publications for their constructive suggestions that have greatly improved the publications.

I made a lot of new friends at DA-IICT, who helped me in many steps of my study. I thank all of them for everything that they did for me. Thanks to Nilay (Jekson-Vision, Ahmedabad), Chintan (Ph.D. candidate at MAASTRO Clinic, Maastricht, The Netherlands), Manali (InFoCusp, Ahmedabad), Naman (NVIDIA, Santa Clara, California), Dr. Jignesh Bhatt (Indian Institute of Information Technology, Vadodara), Dr. Rakesh Patel (Lalbhai Dalpatbhai College of Engineering, Ahmedabad), Dr. Kishor Upla (SVNIT, Surat), Dr. Prakash Gajjar (Government Polytechnic, Surat), Dr. Ashish Phophalia (Indian Institute of Information Technology, Vadodara), other PhD students at DA-IICT Naveen, Sarita, Vandana, Ramnaresh, Shrishail, Kamal, Sonam, Gitam, Sumukh, Parth, Nirmesh, Jainisha,

Madhu, Purvi, Prashant, Harsh and Mahipal for many fruitful discussions. I would like to thank former M.Tech. and M.Sc. students at DA-IICT Jaladhi, Neha, Geethakrishnan, Karthik, Abhijeet, Dhruv, Jai Jai, Chetna, Ananya, Garima, Fenny, Prathmesh, Surabhi, Ketul and Hanish for their kind co-operation. I am also thankful to all my friends from SVNIT, FAMT and VVEHS for their constant support and encouragement. I also wish to express my gratitude towards my friends Abhishek, Reshma and Prachi for their great support and hospitality in the U.K. during the BMVC-2015 conference.

I am grateful to Mr. Dinesh Bhalani and his family for hosting me in Gandhinagar and providing me a *second home*. I would like to thank them for providing me all the help whenever I requested (and many times even before I requested), bearing with me for coming back late at night and for everything they did to make my stay comfortable.

I warmly thank my cousins *Komal* and *Rashmi*, aunts *Shilpa*, *Sujata*, uncle *Dilip* and grandmother *Kumud*. I would also like to express my heartfelt thanks to my late grandfather *Shashikant* for providing me great support and encouragement, not only in academics, but also in extra-curricular activities. He was very happy to see me take up doctoral studies and would had rejoiced to see me graduate. Finally, I would love to cordially express my gratitude, mother *Suniti* whose compassion and determination helped me to become who I am, my father *Gajanan* whose understanding and help led me to where I am, and my uncle *Sudhir* who always encouraged me to take up challenging tasks and bring the best out of me.

Contents

Abstract

Digital reconstruction of ruined historic monuments and heritage sites can help in visualizing how these may have existed in the past. Also, such a process requires no physical alteration to the existing monuments. This facilitates in avoiding their further accidental damage. A digitally reconstructed heritage site in the form of an immersive walkthrough can serve as a delightful tool for both educational and entertainment purpose.

This thesis presents novel approaches for auto-inpainting that involves image inpainting as well as automatic detection of cracks and other damaged regions for inpainting in heritage monuments. As a by-product of one of our inpainting techniques, we are also able to perform resolution enhancement i.e. super-resolution. The purpose is to obtain the digitally reconstructed monuments having enhanced resolution, where the digital reconstruction is performed by automatically detecting and inpainting the damaged regions. The resulting images can serve as an input to immersive walkthrough systems. In our first inpainting approach, we propose an iterative exemplar based method that fills the missing pixels by making use of parameters of an autoregressive (AR) model. These parameters represent the pixel-neighborhood relationship. Considering a set of candidate exemplars, we estimate the parameters of the AR model using the non-negatively constrained least squares (NNLS) method.

In our second inpainting approach, we propose a unified framework to perform simultaneous inpainting and super-resolution. Here, we construct dictionaries of image-representative low and high resolution patch pairs from the known regions in the test image and its coarser resolution. Inpainting of the missing pixels is performed using exemplars found by comparing patch details at a finer

resolution, where self-learning is used to obtain the finer resolution patches by making use of the constructed dictionaries. The obtained finer resolution patches represent the super-resolved patches in the missing regions. Advantage of our approach when compared to other exemplar based inpainting techniques are (a) additional constraint in the form of finer resolution matching results in better inpainting and (b) inpainting is obtained not only in the given spatial resolution but also at higher resolution leading to super-resolution inpainting.

In practice, inpainting techniques perform filling of the missing pixels in known regions i.e., they rely on the user to manually select the regions to be inpainted. This is also the case with our two inpainting methods briefly discussed above. To avoid human interaction i.e., manual selection of region to be inpainted, we propose auto-inpainting techniques wherein the missing regions to be inpainted are not known but are automatically detected prior to inpainting. First, we propose a novel method that detects and inpaints the visually dominant damage regions viz. eyes, nose and lips in facial images of statues. A bilateral symmetry based method is used to identify the eye, nose and lip regions. We then use texton features extracted from each of these regions in a multi-resolution framework to characterize both the damaged and non-damaged regions. For classification, the texton features are matched with those extracted offline from a training set consisting of true damaged and non-damaged regions. If the region is found to be damaged, the best matching non-damaged region from the training set is used to inpaint the identified region using an existing inpainting method.

Having proposed an auto-inpainting technique for facial regions in statues, we proceed to auto-inpainting the non-facial regions in heritage monuments by proposing three techniques for detecting and inpainting cracks. Here, we initially propose a simple pixel based approach that uses order-statistics and density based filters for the detection of cracks. We then propose a patch based approach for crack detection by making use of singular value decomposition (SVD) for patch comparison. Further, we propose another effective and more accurate crack detection method based on comparison of patches using a measure derived from the edit distance, which is a popular string metric used in the area of text mining.

We extend this crack detection approach to perform inpainting of video frames by making use of the scale invariant feature transform (SIFT) and homography. Here, we consider the camera movement to be unconstrained while capturing video of the heritage site, since such videos are typically captured by novices, hobbyists and tourists. Finally, we provide the temporal consistency measure to quantify the quality of the inpainted video.

Keywords:

Inpainting; super-resolution; cultural heritage; digital reconstruction; textons; crack detection; order-statics; SVD; edit distance; SIFT; homography; video quality.

List of Tables

List of Figures

CHAPTER 1

Introduction

Historic monuments and heritage sites across the world are important sources of knowledge which introduce us to our cultural history, depicting the evolution of humankind. They not only represent irreplaceable assets that signify the culture and civilization of the past, but also portray masterpieces of accomplishments that symbolize the human potential. Such places serve as excellent tourist attractions and can contribute significantly to a nation's gross domestic product as tourism is of great economic importance to many industries. It is for this reason many organizations and government agencies globally have been taking keen interest towards safeguarding and preserving the heritage sites.

Over the centuries, the heritage sites have witnessed a number of natural calamities and sabotage resulting in their present ruined condition. A not very distant example is that of the infamous destruction of *the Buddhas of Bamiyan* in Afghanistan by the Taliban as reported in [8, 55]. Fearing the risk of further damage by visitors, access to many heritage sites is now restricted. One such example is that of the *mandapa with musical pillars* in the Vithala temple at Hampi in India, where the visitors are not allowed to touch and experience the chimes resounding from the musical pillars. In order to preserve the heritage sites, one may think of their physical renovation. However, the renovation may not only pose danger to the undamaged monuments, but may also fail to mimic the skillful historic work. Alternatively, it would be interesting to digitally reconstruct the heritage sites since such a process avoids physical contact to the monuments. The digitally reconstructed heritage site in the form of an immersive walkthrough system may then provide an unrestricted access for viewing the monuments in their en-

tirety. This would not only serve as a source for entertainment, but will be a delightful tool for imparting knowledge of history in the form of immersive storytelling. Also, in today's world, with the availability of better computing and storage facilities, preservation of the digitally reconstructed monuments is inexpensive. The digital reconstruction of the heritage sites also facilitates the creation of virtual tours, immersive walkthroughs and mixed-reality systems using digitized 3D models. The *Virtual Asukakyo* project [134] is one such example which digitally restored the ancient capital of Asuka-Kyo in Japan and provided a mixed reality experience to the visitors.

Reconstruction and creation of immersive walkthrough systems using digitized 3D models involves the processing of inputs captured using various acquisition systems viz. laser scanners, multiple photographs for dense reconstruction, etc. Since laser scanners are expensive and usually provide only depth information, one can estimate the same with the help of a large number of photographs (wherein color information is already available) using the technique proposed in [46]. However, in either of the acquisition systems, self-occlusion or difficulty in capturing the scene / object from a particular viewpoint may exist and greater details may not be captured. This results in holes in the captured / estimated 3D surface as well as missing out the high-resolution (HR) details. The holes can be filled up using Poisson surface reconstruction method proposed in [71]. However, in this case the reconstructed surface is over-smoothened and again the HR details are lost.

One can overcome the above problem by filling the missing regions in the captured photographs before estimating the 3D surface. Also, the reconstructed surface retaining the HR details may be obtained by using resolution enhanced version of the photographs. While inpainting can be used to digitally fill or repair the entire damaged region in the photographed scene, the HR details can be obtained using super-resolution. Particularly, when heritage monuments are to be digitally preserved, one needs to bear in mind the desire of viewers to view the monuments in their undamaged form and their excitement for observing fine details of the skillful historic work. By digitally correcting any damaged regions by means

of inpainting and enhancing their resolution using super-resolution, the viewers can be provided with an enhanced visual experience. Thus, the creation of immersive walkthrough systems or digital reconstruction of invaluable artwork consists of image inpainting and super-resolution as the preliminary steps.

The process of selecting the regions to be inpainted is usually subjective. One user may want some region of the image to be modified, while another user may want to modify another region in the same image. Hence, for an inpainting algorithm, the regions to be inpainted are usually selected manually. However, when looking at heritage monuments there is a consensus to view these in their undamaged form. Here, one would crave that the damaged dominant facial regions in statues and cracked regions, which diminish the monument's attractiveness, are automatically detected and corrected in a seamless manner. The exact selected area may vary for different users if the selection of damaged regions in photographs of monuments is done manually. Also, the process of selecting the damaged regions is an enervating task. This necessitates an exploration for a technique that can automatically detect the damage to dominant facial regions in statues and cracked regions in images of monuments, which is a critical and challenging problem for the success of digital reconstruction of heritage sites [29, 34, 35]. An automatic detection of these damaged regions will also facilitate the inpainting to be performed on-the-fly for creating efficient immersive navigation/digital walk-through systems.

This thesis presents novel approaches for auto-inpainting by proposing techniques for image inpainting and automatic detection of cracks & other damaged regions for inpainting in heritage monuments. One of our inpainting approaches also performs simultaneous super-resolution for obtaining higher spatial resolution photographs of the repaired monuments. The resulting images and videos can then serve as an input for 3D surface estimation and eventually for creating immersive walkthrough systems. In what follows, we provide a quick introduction to inpainting and super-resolution followed by a summary of the contents.

1.1 What is inpainting?

Image inpainting is the process of restoring or modifying the image contents imperceptibly. Given an image and a region of interest (ROI) in it, the task of an inpainting process is to fill the pixels in this region, in such a way that either the original content is restored or the region is visually plausible in the context of the image. Image inpainting can be used for a number of applications that require automatic restoration or retouching of some region of a photograph. In fact, the term inpainting is derived from the art of restoring damaged images in museums by professional restorers [9]. Although restoration and inpainting are used interchangeably, inpainting can be considered as a superset of restoration. In general, restoration refers to undoing of degradation, while inpainting also allows creating special effects such as removing / adding objects.

Pixels in the missing region (i.e. the ROI) can be filled either by gradually propagating information from outside the boundary of the ROI or by making use of cues from similar patches. Based on these filling strategies, the existing inpainting methods can be categorized into two important groups viz. (a) methods using level lines and solving partial differential equations (PDEs) and (b) exemplar based techniques. Among these, the exemplar based methods are more popular as processing of similar patches well synthesizes the texture inside the missing regions.

Inpainting is often performed in a semi-automatic manner, in the sense that regions to be inpainted are required to be manually selected by the users. *Blind inpainting* is one of the categories of techniques that do not require any user-interaction for providing the regions to be inpainted. In fact, the techniques under this category perform inpainting without apriori knowledge or detection of the missing pixels. However, they assume the input image to be a noisy observation and perform image recovery by considering all the image pixels to be corrupted by various noise or degradation models. The blind inpainting methods are used when there exists difficulty in selecting the corrupted (i.e. missing) pixels. One may note that the blind inpainting techniques perform well when the missing pix-

els are not localized but randomly spread across the complete image. Moreover, the inpainted regions appear blurred and they fail to recover the texture in large missing regions. Another problem due to the blind nature of such techniques is that the known pixel values also get modified.

A different category of techniques that also facilitate the automatic detection of the regions to be inpainted, is known as *auto-inpainting*. The techniques under this category are developed for digital repair in specialized applications by performing automatic detection and may or may not follow with an implicit method for inpainting the detected regions. Once the regions are automatically detected, it is possible to fill the missing pixels in these regions using a generic inpainting technique. Thus, such methods can avoid the need of manually supplying the regions to be inpainted. Also, unlike blind inpainting the advantage here is that these techniques fill only the detected missing pixels without modifying values of the known pixels.

In this thesis, we also address the problem of super-resolution i.e. resolution enhancement in addition to auto-inpainting since one of the proposed inpainting techniques performs simultaneous super-resolution. Towards this end, a brief introduction to super-resolution is provided in the following section.

1.2 What is super-resolution?

As already discussed, both inpainting and super-resolution are useful in applications involving immersive navigation. Solving of these problems has been attempted separately by researchers. However, part of our work also involves carrying out these operations simultaneously towards providing a unified solution. In this section, we discuss what super-resolution (SR) is, and provide brief details of the major categories of existing SR techniques.

While inpainting is used to fill the missing pixels in the given image, super-resolution (SR) is performed in order to obtain an upsampled version that preserves the high frequency details. A digital image is generated by spatially sampling the continuous scene acquired using an image sensor. If the sampling fre-

quency is low, it introduces distortion due to aliasing in the high frequency components. In addition to aliasing, the sensor point spread function (PSF) and camera motion resulting in optical blurring also degrade the quality of the generated image. Super-resolution refers to an algorithmic approach to obtain a high spatial resolution image from one or more low-resolution (LR) observations, thereby recovering the high frequencies and removing the degradations that arise due to the capture of an image using a camera with low spatial resolution. In effect, the SR process minimizes the aliasing and blurring. In other words, the super-resolved image resembles the true image captured using a HR camera.

Based on the cue used, the existing super-resolution approaches can be classified into two major categories viz. motion-based techniques and motion-free approaches. Motion-based techniques use the relative motion between low resolution observations as a cue in estimating the high resolution image, while motion-free techniques use cues such as blur, zoom, and shading. The later methods do not require observations with relative motion among them. Some researchers have also attempted to solve the super-resolution reconstruction problem without considering any specific cue, but by using a set of training images in order to learn the required information for resolution enhancement. A comparative survey of the motion-based and motion-free SR techniques can be found in the works by Chaudhuri [21], Chaudhuri and Joshi [22], Park et al. [108].

Super-resolution techniques may also be classified depending on the use of the number of distinct observations of an object / scene viz. classical multi-image SR and example-based SR. Here, the techniques that estimate the HR image using multiple LR images of same scene, fall under the classical multi-image SR category. On the other hand, in the example-based SR techniques, the relation between LR-HR patches is learned from a database containing pairs of LR-HR images or from the given image itself. The advantage with example-based SR methods is that these are capable to provide results with high magnification factors. This is unlike the classical multi-image SR approaches which have a limitation on the amount of magnification (approximately twice the input image size) that can be achieved.

1.3 Thesis contributions

Having provided a brief introduction to both inpainting and super-resolution, we now summarize the important contributions of this thesis, the details of which are discussed in the subsequent chapters. We also highlight how these contributions are different from other works.

While image processing and computer vision have been active areas of research and used in different applications, few researchers have attempted the use of image processing techniques in the digital preservation of heritage sites. The book by MacDonald [92] describes the process of digitalization for cultural heritage and provides information about the required hardware and software setup, acquisition of images for 2D and 3D rendering along with related case studies. Likewise, the preservation of original objects at the heritage site or in museums as well as creating their imaged replica in digital form is discussed by Munshi and Chaudhuri [98]. Here, the authors also discuss the issues involved in digital content and community building for archiving and global sharing of heritage resources. In this thesis, we instead address the other aspect of digitizing the cultural heritage viz. reconstruction and recovery of details in the lost or deteriorated regions in the photographs of the monuments, which is not addressed in any of the books mentioned above.

Algorithms to digitally detect and restore typical damages that photographs suffer such as foxing, water blotches, fading and glass cracks are discussed in [129]. They aim to undo any damage to the photograph itself rather than digitally repairing the imaged physical object. On the contrary, this thesis contributes by providing methods for object completion rather than image restoration i.e. our proposed methods perform digital repair of damaged regions and cracks in the photographed monuments, in order to restore the missing details in a heritage scene. Thus, unlike other approaches, the proposed methods detect and repair the defect in the physical object being photographed and not to the photograph itself.

To begin with, we propose two novel inpainting techniques based on the user

supplied regions to be inpainted. Unlike the recent user assisted interactive inpainting, methods proposed in Ghorai et al. [50], Purkait and Chanda [116] that may require special imaging conditions, our proposed methods require no special imaging conditions and aim to perform the digital repair of the photographed physical object / scene. We make use of the artistic details available in the given image itself for completion of the missing regions. By doing so, we are able to imitate the creative expressions of the artists which could be in the form of brush strokes in mural paintings or carvings in petroglyphs and sculptures. One of our proposed methods not only inpaints the given image but also creates the missing details in its higher resolution i.e. super-resolution inpainting. This is particularly suited for digitizing the cultural heritage since both, the inpainting and the super-resolution, are performed simultaneously by the proposed method.

In order to fully automate the inpainting process we also explore techniques for automatic detection of the damaged regions. The exisitng method for detection of defects proposed by Amano [4] works well for detecting computer generated superimposed characters having uniform pattern. A method for pavement crack detection using tensor voting proposed by Zou et al. [160] is heavily dependent on the accuracy of generation of crack-pixel binary map. The method proposed by Cornelis et al. [24] is suitable only for the detection of fine cracks that appear on paintings. Apart from the inpainting techniques, in this thesis we also discuss methods particularly tailored for automatic detection of damaged regions in facial images of statues and cracks in heritage scenes, which to the best of our knowledge have not been attempted previously. Our first auto-inpainting method detects the visually dominant eye, nose and lip regions in statues and then identifies which of these are damaged. We next discuss techniques for detecting cracks and the detected regions are inpainted to obtain the reconstructed view of the photographed monument / scene.

We also extend our work on crack detection to perform auto-inpainting in videos. The video inpainting method proposed in [111] requires the users to manually specify the objects or regions that are to be inpainted. Also, many constraints are placed on the camera movement. Our proposed method for video-inpainting

can detect and inpaint cracks without constraining the movement of the camera. Thus the method is completely automated and needs no human interaction. This can be particularly useful for performing on-the-fly digital reconstruction of damaged regions when tourists capture the heritage monuments using their handheld video capturing devices.

To summarize, we have addressed the problem of auto-inpainting i.e. inpainting along with the automatic detection of the regions to be inpainted. This includes our following works:

- An exemplar based inpainting approach using autoregressive (AR) model,
- A technique for simultaneous inpainting and super-resolution using self-learning,
- A method to identify damaged regions in facial images of statues for inpainting,
- A simple pixel based approach to detect cracks in heritage site images for inpainting using order-statistics and density based filters,
- A patch based technique using singular value decomposition (SVD) to detect cracks for inpainting ,
- A method for more accurate crack detection in heritage site images using tolerant edit distance (tED) for inpainting, and
- An extension to auto-inpaint cracks in videos with quantitative assessment.

1.4 Thesis organization

The contents of this thesis are organized as follows. The literature review is presented in chapter 2. We propose an inpainting technique for filling the missing regions in the object / scene being imaged, in chapter 3. In order to fill the missing regions in heritage scenes by imitating the creative expressions of artists, which could be in the form of brush strokes in mural paintings or carvings in petroglyphs and sculptures, one needs to capture the dependencies in surrounding

regions using pixel-neighborhood relationships. In this chapter, we discuss an iterative exemplar based inpainting technique wherein we use the estimated parameters of an autoregressive (AR) model that represent the pixel-neighborhood relationships. Unlike those approaches which simply copy the pixels to be inpainted from the best matching exemplar, we use the AR parameters in addition to the best matching exemplar to fill the missing pixels so that the artistic creative expressions are retained in the filled regions.

As already discussed, for applications like creating immersive walkthrough systems or digital reconstruction of invaluable artwork both inpainting and resolution enhancement of the given images are the preliminary steps that need to be performed for providing a better visual experience. In chapter 4, we present a unified framework to perform simultaneous inpainting and super-resolution, rather than addressing them in a pipelined manner as is usually done. The presented approach is based on creating pairs of LR-HR dictionaries for self-learning. With the use of this technique, the missing pixels are filled not only in the given spatial resolution but also in the higher resolution leading to super-resolution inpainting.

The inpainting techniques discussed in chapter 3–4 depend on the user to supply the regions to be inpainted. In chapter 5, we discuss a method that automates the process of identifying the damaged dominant regions viz. eyes, nose and lips in face image of statues at a historic site, for the purpose of inpainting. Thus, the missing regions to be filled are not known but is automatically detected prior to inpainting. Here, we use bilateral symmetry of face as a cue to detect the dominant regions followed by matching of texton features to identify the damaged eye, nose and lip regions. Poisson image editing method is then used to inpaint the damaged regions.

The aim of chapters 6–7 is to introduce techniques that can automatically detect the cracked regions in heritage monuments and demonstrate their repair by inpainting. This can be particularly useful for performing on-the-fly digital reconstruction of damaged regions when tourists capture the heritage monuments using their handheld video capturing devices. In chapter 6 we first discuss a pixel based simple method to automatically detect the damaged regions which are char-

acterized by abruptly dark deteriorations in the photographed monuments of a heritage site by making use of of order-statistics and density filters. This is followed by a patch based approach using singular value decomposition (SVD) for automatic detection of the cracked regions in the photographed object / scene, for the purpose of digitally restoring them to their entirety using inpainting. In chapter 7 we discuss another effective and more accurate crack detection method based on comparison of patches using a measure derived from the edit distance, which is a popular string metric used in the area of text mining. We then extend this crack detection approach to perform inpainting of video frames by making use of the scale invariant feature transform and homography. We consider the camera movement to be unconstrained while capturing video of the heritage site, since such videos are typically captured by novices, hobbyists and tourists. Here, we also provide the temporal consistency measure to quantify the quality of the inpainted video.

Finally, in chapter 8 we conclude the thesis by summarizing the main contributions and by listing out future research directions. Most of the material discussed in this thesis has been published in our works [101, 102, 103, 104, 105, 106, 107].

CHAPTER 2

Literature Review

In this chapter, we provide a review of the literature for inpainting and super-resolution highlighting the insights of current research status in these areas. We first review the literature for inpainting in section 2.1 followed by super-resolution in section 2.2.

2.1 Inpainting

Image inpainting has been an active area of research for more than a decade. During the 1990s many researchers addressed the problem of interpolating missing pixel values. However, it was only towards the end of the decade that Masnou and Morel [95] proposed the first inpainting technique. Their method connected the level lines (i.e. curves or contours of constant intensity) arriving at the boundary of an occluded region. With this technique, the occluded regions were filled up but the level lines did not curve in a plausible manner. A major breakthrough to the inpainting problem was later provided by Bertalmio et al. [9]. Their algorithm not only connected the level lines arriving at the boundary of the occluded regions, but also enabled their curving inside the occluded region in a visually plausible manner. The periodical curving of the level lines that also avoids crossings, was achieved using anisotropic diffusion [3, 114]. The algorithm proposed by Bertalmio et al. [9] was successful in propagating structure, however, failed in inpainting the areas having large textured regions. Nevertheless, this method inspired future works that also relied on the propagation of level lines [100, 147].

For propagating texture, a patch replicating method was suggested by Cri-

minisi et al. [25, 26]. This patch-based technique exploits the self-similarity in an image by searching for a similar patch from the surrounding known regions (having no missing pixels) to inpaint the missing pixels in a patch under consideration. Here, the occluded region containing the missing pixels is filled in a patch-by-patch approach by copying pixels from the corresponding similar patches i.e. exemplars. The method emphasizes on the order of selecting the patch to be filled up as this allows the propagation of both structure as well as texture.

Another technique to inpaint large textured regions was proposed by Pérez et al. [112] that used the Poisson's equation for adding objects / texture from other images by using them as guidance vector field. Thus, if a user supplies the region of interest to be edited and a region from which information is to be transferred i.e. guidance vector field, the technique results in a seamless blending of these two regions. Researchers have also proposed methods [54, 151] that make use of level lines for texture synthesis. Yet another patch-based approach is proposed in the work by [149]. This approach fills the missing pixels by considering pixels not only in spatial neighborhood, but also across temporal neighborhood in an iterative multi-scale manner using spatio-temporal pyramids. This is performed by optimizing a cost function that ensure global coherence.

Later, Mairal et al. [93] proposed a technique for color image restoration based on sparse representations obtained by dictionary learning. This method, basically, addresses the problem of learning dictionaries for sparse representation of color images in a quest to extend the K-SVD based denoising approach proposed for grayscale images in [36]. In addition to inpainting, this method is useful in obtaining impressive results for applications like denoising and demosaicing. On similar lines, Mairal et al. [94] proposed a method for image restoration using non-local sparse models. This method provided a framework to combine dictionary learning along with searching for similar patches, using simultaneous sparse coding. Around the same time, Pritch et al. [115] and Barnes et al. [7] proposed fast algorithms for image re-targeting which includes inpainting as one of its applications. This method is non-iterative, which fills the missing regions by mapping these to other locations in the input image by performing graph-cut optimization.

On the other hand, the method proposed in [7] is suited for iteratively filling the missing regions by quickly searching for exemplars using randomized correspondences.

Bugeau et al. [14] proposed a variational model for inpainting that minimizes a cost function involving (a) similarity of patches for texture synthesis, (b) diffusion for propagation of information in the direction of level lines and (c) coherence for maintaining the consistency with neighborhood. Another approach that exploits the redundancy in images for inpainting was proposed by Xiong et al. [153]. This approach fills in the missing regions using self-similar patches in a parameter assisted manner. On similar lines, [80] proposed an inpainting algorithm using a novel strategy for exemplar matching. Here, a fast exemplar search is performed by evaluating matching scores obtained by decomposing the exemplars into frequency coefficients followed by selecting fewer but most significant coefficients. The missing pixels are then filled using the exemplar in gradient based approach.

Li et al. [85] proposed a method based on scene transform and color transfer. Here, a source image that resembles the input image, is searched by matching various features. The missing regions in the input image are then filled using the corresponding regions in the source image by minimizing a cost function involving boundary condition that facilitates plausible blending. Xu and Sun [154] proposed an inpainting method by using sparsity as a cue to assign priority that decides the order in which patches are filled the missing regions. This method prioritizes patches having higher sparsity, while the filling is done by expressing the selected patch as a sparse linear combination of similar patches that are consistent with the neighborhood. Likewise, the method proposed by Huan et al. [61] uses fast marching method to decide the order of filling patches in the missing region, followed by using exemplars and Markov random field (MRF) prior for inpainting these regions.

A different method for inpainting was attempted by Hladky and Pauls [60] by proposing a neuro-biological image completion model. Their method performs completion of occluded visual data by finding solutions to the minimal surface problem with Dirichlet boundary conditions in the roto-translation group. Simi-

larly, the work by Subrahmanyam et al. [130] addressed the problem of inpainting noisy photographs. Their method uses a recursive image recovery scheme for simultaneously inpainting missing regions in an image by using unscented Kalman filter and also suppresses film-grain noise.

Researchers have also attempted inpainting techniques based on probabilistic structure estimation [125], methods using depth [10, 12, 78], focus [96], statistics of patch offsets [59, 76], combining patches from multiple images that may contain same or different objects [28, 150] and enhancement of high frequency details using patch-based anisotropic diffusion [116].

Recent techniques include the works by Ghorai et al. [50], Huang et al. [62], Purkait and Chanda [116], Rematas et al. [118]. While the work by Rematas et al. [118] presents a learning based inpainting technique by making use of the structural information extracted from 3D models a particular class of objects. Filling the missing regions in images containing an object of this class is performed using the learnt structural information. The method proposed by Ghorai et al. [50] performs multiscale image completion by combining transform domain patch filtering. Likewise, Huang et al. [62] proposed a patch-based image completion method that makes use of constraints on patch offsets and transformations derived from the translational regularities in the estimated planes. A comparative survey of inpainting techniques can be found in [56]. However, all the methods discussed above are semi-automatic, in the sense that regions to be inpainted are required to be manually selected by the users.

We now provide a brief overview of the techniques under blind inpainting and auto-inpainting categories, which do not require any user-interaction for providing the regions to be inpainted. However, the amount of published literature under these categories is very much limited, unlike the several techniques discussed above that are supplied with the regions to be inpainted. The blind inpainting techniques proposed in [1, 31, 146, 155] assume the input image to be a noisy observation and perform image recovery by considering all the image pixels to be corrupted by various noise or degradation models. On the other hand, the methods proposed for blind inpainting by Xie et al. [152] and Cai et al. [15]

use sparse stacked denoising autoencoders and convolutional neural networks, respectively. Here, large number of examples consisting corrupted and uncorrupted patch pairs are used for training a model that perform blind inpainting.

Chang et al. [20] proposed a method to auto-inpaint damage in images due to color ink spray and scratch drawing by making use of several filters and structural information of damages. Likewise, the method proposed by Tamaki et al. [133] addresses auto-inpainting string-like objects that block user's view of a discernible scene using contrast as a cue. Amano [4] presented a correlation-based method for detecting defects in images. This method relies on correlation between adjacent patches for detection of defects i.e. small number of regions disobeying an "image description rule", complied by most local regions. Parmar et al. [109] proposed an auto-inpainting technique which uses matching of edge-based features with pre-existing templates to distinguish vandalized and non-vandalized regions in frontal face images of monuments at heritage sites. Similarly, Turakhia et al. [140] proposed a method to automatically inpaint cracks in images of heritage monuments which relies on the detection of edges and tensor voting to identify cracks. Another tensor voting based method for auto-inpainting of cracks in pavement images was proposed by Zou et al. [160] which is heavily dependent on the accuracy of generating the crack-pixel binary map that acts as an input to the tensor voting framework. Later, Cornelis et al. [24] proposed a method for virtual restoration of paintings. This method is flexible as the user may set parameters to suit the input. However, it is suitable only for the detection of fine cracks that appear in paintings.

For inpainting in videos, a method has been proposed by Patwardhan et al. [111]. Their technique considers a static background with a moving foreground, any of which could fall under the region to be inpainted. First, the occluded foreground patches are filled up using motion-inpainting. The background patches which are visible in other frames are then directly copied. Finally, any missing region is filled up using the exemplar-based inpainting approach [26]. It may be noted that, in this approach the users need to manually specify the objects or regions that are to be inpainted. Also, many constraints are placed on the cam-

era motion. Later, Al-Takrouri and Savkin [2] proposed a model based validation technique that uses temporal information to recover corrupted parts of video frames. While a detailed and comparative survey on video inpainting techniques can be found in the works by Ghorai et al. [49], Newson et al. [99], Shih et al. [126], one may note the dependency of these techniques in the robustness of the tracking algorithms used, the lighting conditions and the constraints of camera motion. Most video inpainting techniques can inpaint moving objects under constrained camera motion or a user-selected object. However, to the best of our knowledge, there does not exist any approach that demonstrates video inpainting under unconstrained camera motion with no moving objects and is completely automatic without the need to provide the regions to be inpainted, apart from our proposed technique that we discuss later in chapter 7.

2.2 Super-resolution

The multiple-image super-resolution approach was first addressed in the work by Huang and Tsai [65] which demonstrated the estimation of an HR image using number of LR observations with sub-pixel shifts. Later, Irani and Peleg [66] presented an iterative back-projection based approach wherein the super-resolved image was estimated by iteratively computing the difference between the observed and the simulated LR images from the current SR estimate. An alternate approach which uses l_1-norm minimization and robust bilateral prior was proposed much later by Farsiu et al. [39]. However, by this time a limitation of the classical multi-image SR on the achievable magnification factor was proven as limited ($\approx$2) in the works of Baker and Kanade [6], Lin and Shum [86]. This limitation motivated the exploration for techniques that can learn from examples i.e. *example-based* SR approaches in a quest to attain higher magnification factors.

Freeman et al. [45] were the first to propose an example-based approach for SR. This approach required the construction of an over-complete dictionary consisting of LR-HR patch-pairs obtained from a large set of different training images. Given the test LR image, the generated dictionary was then used to recover the

missing HR details. By placing sparsity constraints on the over-complete dictionaries Yang et al. [156] proposed a technique using the concept of compressive sensing. Here, the SR problem was solved by jointly training LR-HR pair dictionaries and enforcing the similarity of the corresponding sparse representations.

Most of the approaches use some form of smoothness prior that has the tendency to smooth out the textured regions of images for high magnification factors. In order to preserve edges in the super-resolved image, Fattal [40] proposed an approach that used edge dependencies between different resolution versions of an image. Likewise, a gradient profile prior was used in the work by Sun et al. [131], which parametrically defines the shape and sharpness of image edges learned from large number of natural images. Gajjar and Joshi [48] proposed a wavelet learning-based SR approach by making use of detail-preserving *Inhomogeneous Gaussian Markov Random Field* (IGMRF) prior. Here, the learning process provides an initial estimate which is used in estimating the super-resolved image. Later, Freedman and Fattal [44] proposed an approach that uses a non-dyadic filter to preserve the HR image details.

Motivated by the self-similarity properties of images across scales studied by Ruderman and Bialek [121], Turiel et al. [141], researchers also exploited the patch repetitions that occur within-and-across different resolution scales of the given LR image in order to achieve super-resolution. A unified framework combining the classical and example based SR was proposed by Glasner et al. [51], wherein similar LR patches were found within-and across-different image resolution scales. Here, each similar patch to the LR patch imposes a constraint on the corresponding HR patch which is estimated by solving a system of linear equations arising from these constraints. A similar approach was proposed by Luong et al. [90] which uses kernel regression to fuse the similar patches in order to obtain super-resolved image.

Khatri and Joshi [73] drew their inspiration from the work of Glasner et al. [51] and proposed a self-learning algorithm. However, their method builds LR-HR dictionaries comprising patches having exact match with only one coarser-scale version while generating the remaining patch pairs from those learnt using

l_1-minimization. They further improved upon the solution by introducing Gabor prior which forced the similarity of details between LR and downsampled HR patches at various frequencies. The same authors later proposed a method in [74] that removes the redundancies inherent in large self-learned dictionaries to upsample an image without using any regularization methods or priors, which drastically reduced the time complexity involved in performing super-resolution. The most recent approaches for example based SR make use of anchored neighborhood along with learned offline-regressors [136, 137] and transformed self-exemplars [64] which allow geometric variations of the example patches in order to cover significant number of textural appearance variations in the scene.

Recently, deep learning based approaches have provided state-of-the-art results in many areas, including super-resolution imaging. These methods [27, 32, 33, 87, 148] learn a cascade of filters and use them to enhance the spatial resolution of an input image. It is interesting to note that a sparsity prior is used in most of these methods to learn the cascade of filters.

As we have seen above, both the research areas viz. inpainting and super-resolution have been well-explored in the last two decades. Yet, little work has been reported in the literature that provides a unified approach for solving these problems. A method for resolution enhancement using total variation inpainting model was attempted by Chan et al. [18]. However, this method required several LR versions that were blurred, noisy and containing missing pixels information in order to obtain the super-resolved version. Later, Bhavsar and Rajagopalan [11] proposed a joint method for inpainting and super-resolution of range maps having missing data. The same authors also proposed another method for super-resolution and inpainting of range maps using multiple relatively shifted LR range images. Around the same time, Le Meur and Guillemot [81] proposed a method to perform inpainting based on super-resolution. This method however, performs inpainting at a coarser resolution followed by independent super-resolution to get the inpainted image at the original resolution.

Having discussed the current research status in inpainting and super-resolution, we present our first inpainting approach in the following chapter 3.

CHAPTER 3

An Exemplar based Inpainting using Autoregressive Model

When capturing a photograph, some parts of the scene / object may not be visible due to self-occlusion or difficulty in capturing the scene / object from a particular viewpoint. Similarly, in case of heritage monuments, it may happen that the monument is ruined or a part of it is damaged, leaving behind cracks. Such situations result in creation of holes i.e. missing information. In this chapter, we present an inpainting method i.e. a process of restoring the image contents imperceptibly, to fill the missing regions in captured the photographs. Given the region to be inpainted, our method makes use of similar patches i.e. exemplars to fill the missing pixels. Note that this work provides an inpainting technique but does not constitute automatic detection since the regions to be inpainted are provided as an input. Nevertheless, the proposed technique can also be used for inpainting automatically detected regions. A set of exemplars is automatically searched considering a window around every pixel in the region to be inpainted. The sum of squared differences (SSD) criteria is used to determine the similarity of patches. The novelty of our technique lies in the use of parameters of an autoregressive (AR) model that are estimated using the non-negatively constrained least squares (NNLS) method [23].

Unlike simply copying into the target i.e. pixels to be inpainted, values from the best matching exemplar as is done by many of the exemplar based methods [26, 113, 149], we use the AR parameters in addition to the best matching exemplar to fill the missing pixel values. For a set of candidate exemplars, the AR

parameters suggest the contribution of values of neighboring pixels towards the respective center pixel of every 3×3 region in that set. A good source for filling the missing pixels (i.e. an exemplar) may not always be available in the image. In such situations, estimating a pixel value by making use of (a) knowledge of spatial relationship of pixels and (b) information from an exemplar, avoids the seam that may arise due to direct copying of pixels from the exemplar. We briefly discuss the limitations of existing approaches in section 3.1 and describe the details of our proposed algorithm in section 3.2. The experimental results are illustrated in section 3.3 with the help of true images captured from the world heritage site Hampi, Karnataka, India. Section 3.4 concludes the chapter.

3.1 Limitation of existing approaches

Criminisi et al. [25, 26] proposed an inpainting technique that makes use of block replication (i.e. example patches or exemplars) to fill the missing pixels. This technique enabled the propagation of texture inside the missing regions that was earlier not possible with the methods that used partial differential equations (PDEs) [9, 95]. The exemplar based method exploits the fact that for a small patch at the boundary of the missing region, a similar patch i.e. an exemplar can be found from the surrounding known region in the given image itself. As shown in figure 3.1, the missing pixels in this patch are then filled by copying corresponding pixels from the exemplar. This method emphasizes on order of selecting the patch to be filled up, which allows the propagation of structure as well as texture. However, simply copying of the pixels results in visible seam where even the most similar exemplar is considerably dissimilar from the patch to be filled.

Another technique wherein objects / texture from either (a) the input image or (b) different images can be a source for inpainting was introduced by Pérez et al. [113]. Here, the source regions provide the guidance vector field for estimating the missing pixel values. Thus, if a user supplies a region of interest to be edited and a region from another (or same) image from which information is to be transferred (guidance vector field), a seamless blending is achieved by solving for the

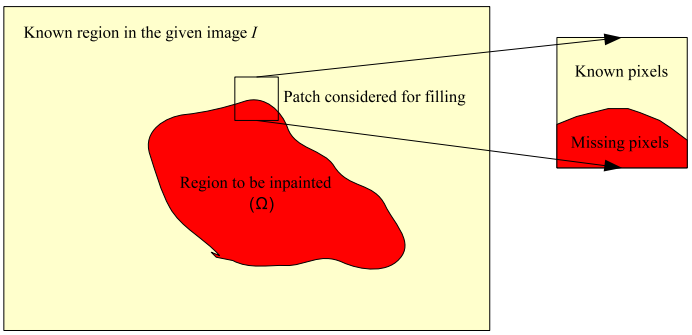


Figure 3.1: Setup for exemplar based image inpainting

Poisson equations. However, the guidance vector field needs to be manually selected leading to highly subjective results. Moreover, the objective function to be minimized determines the inpainted pixel values as the average of neighboring pixel values, which is not true for all images. This motivates us to use an AR model based approach for inpainting.

3.2 Proposed approach

Our work addresses the limitations mentioned in section 3.1 of the widely used inpainting algorithms. Because of the spatial dependency of a pixel value on its neighbors, an AR model can be used to express this dependence, where a pixel value is a linear combination of values of its neighboring pixels [69]. Considering a first order neighborhood, we make use of a set of exemplars to estimate the AR parameters. These parameters in addition to the best matching exemplar are used as constraints for estimating the values of the missing pixels. Since the AR parameters suggest the contribution of neighbors, a non-negative least squares (NNLS) method [23, 79] is used to calculate the values of these parameters. A simple least squares (LS) method is unsuitable for this purpose as it may lead to negative values for the AR parameters, which do not correspond to the correct fractions.

An outline of our proposed method is shown in figure 3.2. We start with an input image I and a user defined mask that specifies the region to be inpainted or the target region Ω as shown in figure 3.1. Let the boundary of Ω be denoted by $\delta\Omega$. Our method makes use of the approach in [26] to determine the patch

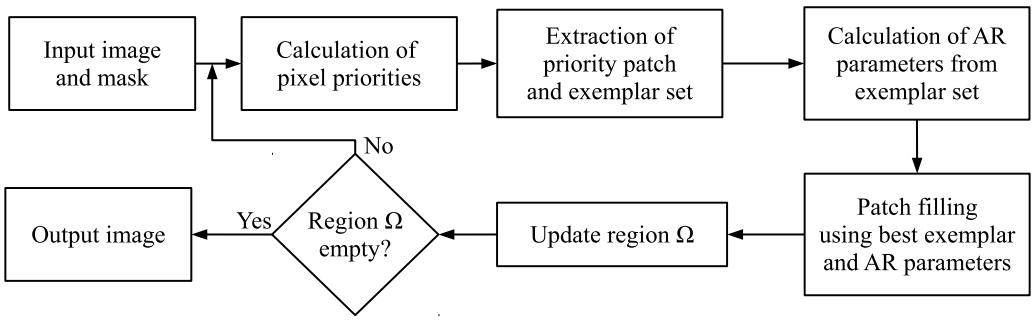


Figure 3.2: Proposed approach.

priority for inpainting, wherein for every pixel $p \in I$ a confidence term $C(p)$ and data term $D(p)$ are calculated. Initially, the confidence term (initial confidence $c(p)$) is calculated as:

$$c(p) = \begin{cases} 1, & \forall p \in I - \Omega \\ 0, & \forall p \in \Omega \end{cases} \tag{3.1}$$

Now considering a fixed size patch Ψ_p around every pixel p, the confidence term $C(p)$ and the data term $D(p)$ are calculated as:

$$C(p) = \frac{\Sigma_{q \in \Psi_p} c(q)}{|\Psi_p|}, \quad D(p) = \frac{|\nabla I_p^{\perp} . \mathbf{n}_p|}{\alpha}, \tag{3.2}$$

where $|\Psi_p|$ is the area of patch Ψ_p around pixel p, $\nabla I_p^{\perp}$ is orthogonal to the gradient ∇I_p at a pixel p, $\mathbf{n}_p$ is unit normal to the target boundary $\delta\Omega$ and α is the normalization factor taken to be 255 (for grey-level images). The priority $P(p)$ associated with every pixel p in $\delta\Omega$ is given by:

$$P(p) = C(p)D(p). \tag{3.3}$$

Once these priorities are calculated, the patch $\Psi_{\hat{p}}$ around the pixel $\hat{p}$ having maximum priority is considered for filling.

Since the pixels on the boundary of region Ω to be inpainted get more priority, the selected patch $\Psi_{\hat{p}}$ will always consist of both the known and missing pixels (as shown in figure 3.1). We intend to make use of information from a similar patch i.e. an exemplar to fill the missing pixels in patch $\Psi_{\hat{p}}$. An exemplar can be

found by comparing the patch $\Psi_{\hat{p}}$ with a patch $\Psi_{\hat{q}}$ around every pixel $\hat{q}$ in the entire image, but as the image size increases the time taken to find an exemplar also increases. However, we observe that a patch is similar to those in its surrounding region. We therefore restrict the search for matching the patches to a large sized window $W_{\hat{p}}$ around the patch to be filled up instead of searching the whole image. By doing so, the number of computations required to search the exemplar are considerably reduced. We measure the similarity of patches $\Psi_{\hat{p}}$ and $\Psi_{\hat{q}}$ by comparing the pixels $\hat{p}_i$ in patch $\Psi_{\hat{p}}$ that are known (i.e. $\hat{p}_i \notin \Omega$) with corresponding pixels in every patch $\Psi_{\hat{q}}$ (where $\Psi_{\hat{q}} \in W_{\hat{p}}$ and $\Psi_{\hat{q}} \cap \Omega = \phi$). The patch $\Psi_{\hat{q}}$ which gives the minimum sum of squared difference (SSD) is considered as the exemplar $E_{\hat{p}}$.

Once the exemplar $E_{\hat{p}}$ is available, one may be tempted to fill the missing pixels by simply copying the corresponding pixels from the exemplar $E_{\hat{p}}$ as done in [26]. This works well if the pixel intensities in the window $W_{\hat{p}}$ do not vary much, leading to small SSD error. However, it may be noted that due to variation in illumination or contrast within the window $W_{\hat{p}}$, which is often the case for images of heritage scenes, the SSD obtained for $E_{\hat{p}}$ will be high, i.e. even the most similar patch will also be considerably different from $\Psi_{\hat{p}}$. In such cases, if pixel values are copied from the $E_{\hat{p}}$ into $\Psi_{\hat{p}}$, the modifications in $\Psi_{\hat{p}}$ do not appear to be uniform, making the seam clearly visible in the inpainted patch. In order to get better inpainting, one can think of using pixel-neighborhood relationship for filling the missing pixels in the patch $\Psi_{\hat{p}}$.

For estimating the pixel neighborhood-relationship, we model the central pixel value of a 3×3 region to be a linear combination of the values of its first order neighboring pixels. Let k_t, k_r, k_b and k_l denote the contributions of the top, right, bottom and left neighbors, respectively towards the central pixel value of a 3×3 region. We now arrange all the patches $\Psi_{\hat{q}} \in W_{\hat{p}}$ in ascending order of the SSD error and consider only the first L patches to form a set $S_{\hat{p}}$ for estimating these contributions. Note that for a small patch (say of size 9×9) the pixel-neighborhood relationship within the patch is more or less consistent. Arranging the patches in ascending order of SSD gives the candidate exemplars that are most similar to the patch $\Psi_{\hat{p}}$ to be filled. Thus, the coefficients associated with the neighboring pixels

would remain consistent of all 3×3 regions inside small similar patches, i.e., $S_{\hat{p}}$, and we can therefore write

$$g_p = k_t g_{q_t} + k_r g_{q_r} + k_b g_{q_b} + k_l g_{q_l}. \tag{3.4}$$

where g_p is the central pixel value of every 3×3 region in $S_{\hat{p}}$ and $g_{q_t}, g_{q_r}, g_{q_b}, g_{q_l}$ are the top, right, bottom and left neighboring pixel values, respectively. A similar method is used in [69] for enhancing the resolution of multispectral images using an autoregressive model. The equation (3.4) represents the equation for an AR model [47] for the pixel g_p. Now that the values of g_p and corresponding first order neighbors are known by considering the set $S_{\hat{p}}$, we estimate the values of k_t, k_r, k_b and k_l that best fit the AR model.

Using least squares (LS) method to determine these AR parameters may result in few of them being negative. However, since the coefficients associated with neighboring pixels denote the proportion of the respective neighbor's contribution, a negative value is unacceptable. We therefore use the NNLS method [23, 79] to obtain the values of k_t, k_r, k_b and k_l, which assures that the obtained values are non-negative. The NNLS method iteratively categorizes the constraints into *active* and *passive* sets. The constraints corresponding to a negative or zero regression coefficient are included in the active set and the remaining constraints constitute the passive set. The solution then corresponds to unconstrained least squares solution using only the variables corresponding to the passive set by setting the regression coefficients corresponding to the active set to zero.

The number of exemplars L in the set $S_{\hat{p}}$ need to be greater than or equal to the number of AR parameters to be determined, else we are left with more number of unknowns to be estimated from lesser number of constraints. Further, as L increases, we have more number of constraints which generalize the model leading to better estimates of AR parameters. However, if L is very large, then many exemplar candidates with larger SSD values (i.e. outliers) get involved in calculating the AR parameters. As a result the estimated values do not represent the true spatial relationship of the pixels in the patch $\Psi_{\hat{p}}$. Therefore, one has to heuristically choose the number L. One may consider filtering the outliers instead of setting

a particular value for L. However, the problem with SSD is that patches that are visually similar can also have higher SSD, say due to spatial shifting of pixels in the two patches being compared (as shown later in figure 4.2). In such cases too, the first-order pixel-neighborhood relationship is helpful for estimating the AR parameters and we therefore need to consider such patches by setting a particular value for L instead of thresholding the outliers based on SSD. In case a patch has many similar patches, are already considering the L best candidate exemplars by arranging the patches in the ascending order of SSD. Considering more number of similar patches will not significantly vary the estimation of the AR parameters, but instead will add to the computational overhead.

One has to also take care that the size of patch $\Psi_{\hat{p}}$ is not very large. If a patch with large size is considered, the size of patches in the set $S_{\hat{p}}$ (that are used to calculate the AR parameters) will also be large. Over a large region, the spatial relationship of the pixels may change and the spatial relationship may not be accurately represented by the AR parameters, and in turn reducing the effectiveness of the algorithm. For images of heritage sites, this is of particular importance for re-creating fine artistic details over small local regions.

With the availability of the exemplar $E_{\hat{p}}$ and the pixel-neighborhood relationship in the form of AR parameters, we estimate the missing pixel in $\Psi_{\hat{p}}$ by blending the corresponding pixels from the exemplar. For this purpose, we use a method derived from the work by Pérez et al. [113], which demonstrates the seamless blending of pixel values from a source region in one image into a missing region in the same or a different image, by solving for unknown pixel values using discrete Poisson equations. In their work, the value of every pixel p in the missing region Ω satisfies the following equation:

$$|N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \delta\Omega} f_q^* + \sum_{q \in N_p} v_{pq} \tag{3.5}$$

where f_p is a value of pixel $p \in \Omega$, f_q^* is a value of pixel $q \in I - \Omega$, f_q is a value of pixel $q \in N_p \cap \Omega$, N_p is the set of neighbors of pixel p, $|N_p|$ is the number of pixels in the set N_p. Here v_{pq} denotes the difference between values of a pixel p

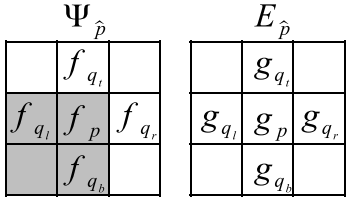


Figure 3.3: $\Psi_{\hat{p}}$ is the patch considered for filling, $E_{\hat{p}}$ is the corresponding exemplar. The shaded region denotes the pixels in the missing region Ω. f_p is the value of pixel p in $\Psi_{\hat{p}}$, g_p is the corresponding pixel's value in the exemplar $E_{\hat{p}}$.

and its neighboring pixel q in source region which correspond to the pixel p in the missing region and its neighboring pixel q, respectively. One may note that in [113] both the source and missing regions are selected manually since the technique is primarily for image editing. Also, the blending is done considering all the missing pixels at once and not in a patch based approach. In our technique, the automatically obtained exemplar corresponding to the patch selected for filling is considered as the source region. If we re-arrange equation (3.5) we get,

$$f_p = \frac{\sum_{q \in N_p \cap \Omega} f_q + \sum_{q \in N_p \cap \delta\Omega} f_q^* + \sum_{q \in N_p} v_{pq}}{|N_p|} \tag{3.6}$$

Considering this scenario in a patch based approach as shown in figure 3.3 and taking $|N_p| = 4$, the equation (3.6) can be be written as:

$$f_p = \frac{(f_{q_b} + f_{q_l}) + (f_{q_t} + f_{q_r}) + (4g_p - (g_{q_t} + g_{q_r} + g_{q_b} + g_{q_l}))}{4} \tag{3.7}$$

where $\sum f_q = f_{q_b} + f_{q_l}$, $\sum f_q^* = f_{q_t} + f_{q_r}$ and $v_{pq} = g_p - g_q$. The above equations (3.6) and (3.7) show that the value f_p is determined by taking the average of its neighbors and by computing the average in the corresponding source patch selected manually. However, for many images, it is not true that a pixel value is an average of its neighbors. In general, pixel value can be considered as a linear combination of the first order neighboring pixel values, and for this reason we modify the equation (3.7) as,

$$f_p = (k_t f_{q_t} + k_r f_{q_r} + k_b f_{q_b} + k_l f_{q_l}) + g_p - (k_t g_{q_t} + k_r g_{q_r} + k_b g_{q_b} + k_l g_{q_l}). \tag{3.8}$$

Here k_t, k_r, k_b and k_l are the estimated AR parameters representing the contributions of each first-order neighbor as shown in figure 3.3 while the exemplar $E_{\hat{p}}$ provides the guidance vector field. It is worth to note that when $k_t = k_r = k_b = k_l = \frac{1}{4}$, equation (3.8) reduces to equation (3.7). The missing pixel values f_p are now estimated by posing the optimization problem as follows:

$$\min \sum_{\forall f_p \in \Psi_{\hat{p}} \cap \Omega} \left|\left| f_p - \begin{pmatrix} (k_t f_{q_t} + k_r f_{q_r} + k_b f_{q_b} + k_l f_{q_l}) + \\ g_p - (k_t g_{q_t} + k_r g_{q_r} + k_b g_{q_b} + k_l g_{q_l}) \end{pmatrix} \right|\right|^2 . \tag{3.9}$$

Once a patch is processed, its pixels are excluded from the region to be inpainted Ω. The updated missing region is then used in the next iteration and after each iteration the missing region Ω shrinks. The algorithm terminates when all the missing pixels are filled.

3.3 Experimental results

We now present the results of our technique on data collected from the world heritage site, Hampi, Karnataka, India. These images were captured using a Samsung ES55 digital camera. The data consists of a number images of monuments, having both damaged and non-damaged regions. The experimental results for three such images are shown in figures 3.4(a), 3.5(a) and 3.6(a). In all the images, fairly large cracks are visible and the aim is to restore the images as if they had no cracks at all.

We show a comparison of the results of our proposed algorithm with that of the algorithm presented in [26]. Both the algorithms are implemented in Matlab. Once an image is given as an input, the user selects the region to be inpainted i.e. Ω that has to be filled. These regions selected by volunteers to be inpainted are shown in black color in figures 3.4(b), 3.5(b) and 3.6(b), respectively. The results using algorithm in [26] are shown in figures 3.4(c), 3.5(c) and 3.6(c), respectively, and those of the proposed method are displayed in figures 3.4(d), 3.6(d) and 3.6(d), respectively. A patch size of 9×9 was selected, the window $W_{\hat{p}}$ is chosen to be of size 37×57. In our experiment we consider the number of patches in

(a) Input image

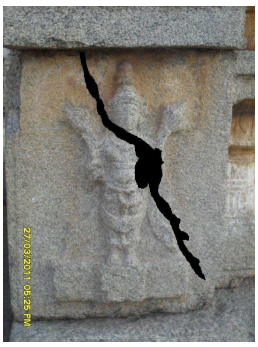
(b) User selected region to be inpainted shown in black color

(c) Inpainted result using method proposed in [26]

(d) Inpainted result using our proposed method

Figure 3.4: Result showing the inpainting of a crack in a wall carving.

(a) Input image



(b) User selected region to be inpainted shown in black color

(c) Inpainted result using method proposed in [26]

(d) Inpainted result using our proposed method

Figure 3.5: Result showing inpainting of a long crack with varying width across a stone-work.

the set $S_{\hat{p}}$ used for estimating the AR parameters to be $L = 30$. The search for exemplars is performed by comparing the respective color channels of the patches

(a) Input image (b) User selected region to be inpainted shown in black color (c) Inpainted result using method proposed in [26] (d) Inpainted result using our proposed method

Figure 3.6: Result showing inpainting to a narrow damaged portion of a statue.

under consideration. Likewise, when transferring information from the exemplars to the missing pixels, all the color channels are processed one by one. This enables color inpainting, which is an interesting feature for heritage images. Alternatively, the exemplar search can be performed in the YCbCr color space by comparing the luminance channel of the patches, while the filling of the missing pixels can be performed by processing each color channel.

Since the reference images are not available, the two techniques cannot be compared using the standard criteria for quantitative comparison viz. like peak signal-to-noise ratio (PSNR), root mean squared error (RMSE), structural similarity (SSIM) index, feature similarity (FSIM) index, etc. Nevertheless, in the areas of image, compression, fusion and restoration researchers have used the entropy, standard deviation and local mean orientation dominance measures to quantify the image quality when no reference images are available [42, 127, 132, 159]. We therefore consider these measures to compare the inpainted results shown in figures 3.4–3.6. Here, we also consider the Blind/Referenceless spatial Image QUality Evaluator (BRISQUE) proposed in [97] to measure the naturalness of inpainted images, as well as the inpainted image quality assessment (IIQA) measure proposed in [139] for comparing the inpainted results. We provide a brief description of these measures below.

The first measure, entropy (used in [159] for evaluation the quality of image fusion), quantifies the information contained in an image. More the entropy, better

is the resultant image. Entropy H is calculated as:

$$H = -\sum_{i=0}^{G-1} p(i) log_2(p(i)) \tag{3.10}$$

where $p(i)$ is the probability or the normalized histogram of i^{th} gray level in an image having G gray levels. A high value of entropy indicates higher value of contrast between adjacent pixels, which is a typical characteristic of porous surfaces found in heritage monuments. The second measure i.e. standard deviation (SD) [127] also suggests about the contrast in an image. For an image having high contrast, the value of SD is high. With the same notations used for calculating the entropy in equation (3.10), the standard deviation SD is calculated as follows,

$$SD = \sqrt{\sum_{i=0}^{G-1} (i - i')^2 p(i)} \quad \text{where,} \quad i' = \sum_{i=0}^{G-1} ip(i). \tag{3.11}$$

The third measure i.e. local orientation dominance R_j indicates the presence of an edge in a local region j [42, 132]. We have considered the local region j to be of size 5×5. R_j is calculated using singular values s_1 and s_2 of the local gradient vectors as,

$$R_j = \frac{s_1 - s_2}{s_1 + s_2}, \quad s_1 \geq s_2, \quad 0 \leq R_j \leq 1. \tag{3.12}$$

It may be noted that for a noisy image, the mean value $\overline{R} = avg(R_j)$ is low, while the same is high for an image with better visual quality.

BRISQUE [97] tries to measure the naturalness of an image. Here, the cue used is that the normalized luminance coefficients of natural images closely follow Gaussian-like distribution. A normalized luminance image is obtained by considering the mean and the standard deviation of 3×3 neighborhood weighted using a 2D circularly symmetric Gaussian function around every pixel. The pair wise products of each pixel (i.e. normalized luminance coefficient) along the horizontal, vertical, main-diagonal and secondary-diagonal directions are then used to plot histograms. For each of these histograms the shape, mean, left variance and

right variance along with the shape and variance of the histogram of the normalized luminance image are used to obtain 18 features. This process is repeated at a coarser scale to obtain a total of $18 \times 2 = 36$ dimensional feature vector which is then classified based on trained classes of feature vectors. A score based on correlation with these classes is taken as the BRISQUE score, where the best and worst qualities are represented by values 0 and 100, respectively. The IIQA measure [139] tries to quantify the quality of the inpainted image based on the coherence of the inpainted regions with the rest of the image and a saliency based structure term. The coherence term captures the similarity of every inpainted patch with its closest match in the rest of the image while the saliency-structure is generated using a mean and Gaussian blurred version of a neighborhood around every pixel. The IIQA measure is then calculated by summing the product of these terms over all the inpainted pixels and normalizing the resultant value. Higher the value of IIQA measure, better is the quality.

The quantitative comparison for the results illustrated in figures 3.4–3.6 using the quality measures discussed above, is shown in table 3.1. From this comparison one can observe that for the results shown in figure 3.4, our proposed method performs better than the approach in [26] for all the quality measures. However, for the results shown in figure 3.5, the proposed method performs better in terms of entropy, standard deviation and local orientation dominance, but not in terms of BRISQUE and IIQA. Similar ambiguity is observed from the quantitative comparison of the results shown in figure 3.6, where our proposed method performs

Images	**Entropy (H) [159]**			**Standard deviation (SD) [127]**			**Avg. local orientation dominance ($\overline{R}$) [42, 132]**			**BRISQUE [97]**			**IIQA [139]**		
	A	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
Figure 3.4	7.4746	7.5435	7.5939	45.8524	47.9063	47.9363	0.2000	0.2220	0.2224	10.6262	11.8215	11.7264	0.5084	0.3278	0.3338
Figure 3.5	6.8010	6.8435	6.8473	29.5197	30.3779	30.4150	0.2583	0.2428	0.2432	31.1195	32.2848	32.3708	0.2331	0.2334	0.2275
Figure 3.6	7.0230	7.0424	7.0426	31.7438	32.1608	32.1703	0.3395	0.2938	0.2491	44.2903	30.1808	30.0610	0.1861	0.1529	0.1193

Table 3.1: Comparison in terms of entropy, standard deviation, average local orientation dominance, BRISQUE and IIQA quality measures for the results shown in figures 3.4–3.6. Here, **A** represents the input image to be inpainted, **B** denotes the approach in [26] and **C** indicates the proposed approach.

better in terms of entropy, standard deviation and BRISQUE, but not in terms of local orientation dominance and IIQA. This makes it difficult to suggest which of the two methods performs better inpainting.

The problem in evaluating the quality of inpainted images is that, the ground truth is seldom available for comparison and one does not know if the inpainted region represents the true missing pixel information. Even if the ground truth image is available, the inpainted region can be completely different from the ground truth and yet appear visually plausible. This makes it difficult to have an objective comparison of the inpainting algorithms. To the best of our knowledge and as suggested in the literature [5, 138, 139], the best way to evaluate the quality of image inpainting is to assign a subjective score with the help of human observers. We therefore use the following method for comparing our inpainting results.

The inpainted results were displayed side-by-side to 4-5 volunteers. Here, the input images and the regions to be inpainted were not shown to the volunteers in order to avoid any bias about the plausiblity of the inpainted regions. The volunteers were then asked to rank the images according to their naturalness. Unless any of the results appeared extremely synthetic, this step usually resulted in an ambiguous answer that all were natural. After this step the input images and the region to be inpainted were also shown to the volunteers and they were asked to update their rankings if required and provide a reason for the rankings. A consensus of the reasons and the corresponding ranks were then used to judge the best inpainted results. Based on this method, a qualitative comparison of the results shown in figures 3.4–3.6 is given below.

Observe the inpainted area below the knee of the dancer in the central region in figure 3.4(c). A clear seam is visible in the inpainted area using the method proposed in [26]. On the other hand, the corresponding region shown in figure 3.4(d) has been seamlessly inpainted using the proposed method. In figure 3.5(c) a repetitive pattern is visible in the inpainted hand of the stone-work along with a seam on the stomach. On the contrary, in the corresponding regions shown in figure 3.5(d) using the proposed method, the inpainting appears to be plausible and seamless. In the inpainted result shown in figure 3.6(c) that makes use of

the technique proposed in [26], a clear contrast in color can be seen inside the inpainted region. There is no such color contrast in the inpainted result of our proposed approach as seen figure 3.6(d). Here, pixels inside the inpainted region exhibit an effectual blending due to which no seam is visible.

Thus, from the figures 3.4–3.6 one can notice the seam at the inpainted regions of the images in figures 3.4(c), 3.5(c), 3.6(c), respectively, obtained using the technique proposed in [26]. At the same time, the results obtained using our technique shown in figures 3.4(d), 3.5(d) and 3.6(d), respectively, are seamless and plausible.

3.4 Conclusion

We have presented an automatic exemplar search based inpainting technique in this chapter. The spatial dependence of a pixel with its neighbors is used here as the cue to blend information from the exemplar into the missing pixels of the patch under consideration. Assuming the neighborhood to be of the first order, the spatial dependence is represented using an AR model, the parameters of which are estimated from a set of candidate exemplars using the NNLS method. Proposed method avoids the direct copying of pixels from the exemplar as this results in visible seam, instead it uses the estimated AR parameters and the exemplar to perform a seamless blending. As seen from the experimental results, the reported results are promising. We conclude that by estimating the spatial dependence of a pixel with its neighborhood using the AR model, the damaged regions in images of monuments can be inpainted plausibly. The images of heritage site inpainted in this manner will provide better inputs for estimating image based 3D models.

CHAPTER 4

Simultaneous Inpainting and Super-resolution

In chapter 3 we discussed an image inpainting technique that fills the missing pixels by using the pixel-neighborhood relationship in exemplars. In this chapter, we discuss a method that not only inpaints the manually selected missing region but also performs super-resolution (SR). Past two decades have seen significant advancement in the techniques for inpainting and super-resolution. Although both problems involve the searching and processing of similar patches for estimating the unknown pixel values, the two problems have been addressed independently. As already explained in chapter 1, both the inpainting and the resolution enhancement i.e. super-resolution, can be used as preliminary steps for creating 3D models in applications like immersive walkthrough systems. However, the usual practice is to solve these two problems independently in a pipelined manner i.e. first inpaint and then enhance the spatial resolution. This chapter provides a unified framework to perform simultaneous inpainting and super-resolution.

In this approach, we construct dictionaries of image-representative low and high resolution (LR-HR) patch pairs from the known regions in the test image and its coarser resolution. The inpainting of missing pixels is then performed using exemplars that are found by comparing patch details at a finer resolution. These patches represent the higher resolution patches in the missing regions and we obtain them from the constructed dictionaries by using self-learning [73]. Here, the advantages when compared to other exemplar based inpainting techniques are (a) the constraint in the form of finer resolution patch matching results in good exemplars and better inpainting, and (b) inpainting is obtained not only in the given spatial resolution but also at higher resolution leading to super-resolution

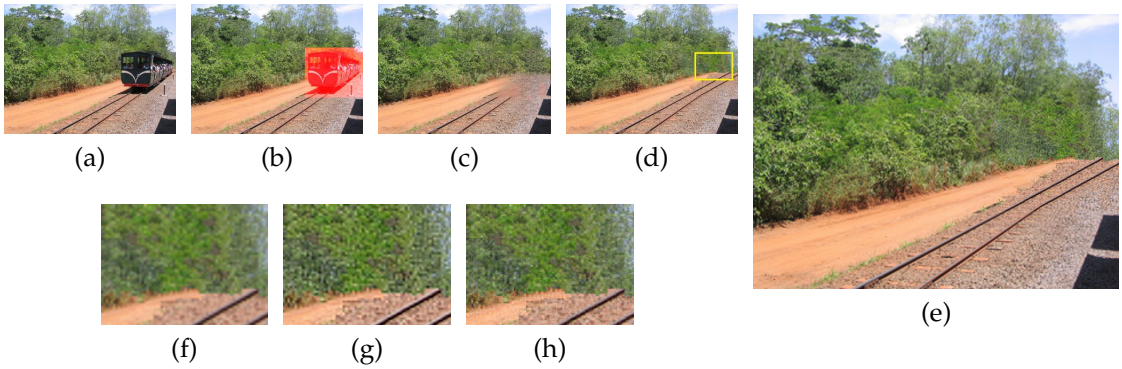


Figure 4.1: Simultaneous inpainting and super-resolution: (a) input; (b) region to be inpainted marked in red color; (c) inpainting using planar structure guidance [62]; (d) inpainting using proposed method indicated by a region inside the rectangular area with a yellow boundary; (e) simultaneously inpainted and super-resolved image (by a factor of 2) using the proposed method with known regions upsampled using bicubic interpolation; (f)–(h) zoomed versions after upsampling (inside region marked by the rectangular area with a yellow boundary in (d)) using various approaches viz. (f) bicubic interpolation, (g) Glasner et al. 's method [51] and (h) proposed method for super-resolution.

inpainting. In other words, we obtain SR as a consequence of inpainting, thus reducing the number of computations when compared to performing these operations independently.

Note that our approach performs super-resolution without introducing blur or artefacts indicating better inpainting at the given resolution. Also, our method does not use any kind of regularization as used by most of the super-resolution approaches [117, 158]. We once again emphasize that the primary goal here is to obtain better inpainting. The super-resolution is obtained as a by-product since we use a constraint that helps in finding a better source for inpainting. One of the results of this method for a natural image is shown in figure 4.1.

The contents of this chapter are organized as follows. In section 4.1 we discuss the need for comparing patches at finer resolution. Our proposed approach is discussed in section 4.2. The efficacy of this method in comparison to state-of-the-art methods is illustrated by showing results on natural images in section 4.3 where we also present the results for image captured at heritage sites. The chapter ends with the conclusion in section 4.4.

4.1 Need for patch comparison at finer resolution

Before we enter into the discussion of our approach for simultaneous inpainting and super-resolution, we would like to point out the need for comparing patches at finer resolution. Natural images including those of heritage scenes usually contain many self-similar patches. This cue has been used effectively by exemplar based inpainting methods [26, 149], where search is done for the region to be filled. However, when similar patches are unavailable, the inpainting may not be seamless resulting in graphical garbage. Even when similar patches are available, the best match may not always be a good source for inpainting. The reason is that the patch to be filled has too little number of known pixels to obtain a reliable match. One may increase the patch size to have more number of known pixels. However, we may not find good matches for larger patches due to which the inpainted regions look implausible.

In exemplar based inpainting approaches, patch matching is done by discarding the missing pixels. Due to this, it may happen that a better source patch for inpainting could be found among the patches other than the best match as illustrated in figure 4.2. It is desirable to consider these patches as candidate sources for inpainting without discarding them. Intuitively, by performing a detailed assessment of the patches to be filled, one can confidently determine which among the candidates is a better source for inpainting. In other words, if the high-

y_p y_{q_1} (SSD = 48821) y_{q_2} (SSD = 49131) y_{q_3} (SSD = 49597) y_{q_4} (SSD = 54113) y_{q_5} (SSD = 54495)

Figure 4.2: Matching patches in exemplar based approaches considering the sum of squared distance (SSD) measure. Here, y_p is the patch selected for inpainting with the missing pixels shown in red color. The patches $y_{q_1}, \ldots, y_{q_5}$ are the most similar patches to y_p in terms of SSD. Although y_{q_1} has a smaller value for SSD, we observe y_{q_2} and y_{q_3} to be better sources for filling the missing pixels in y_p. Note that red color indicates region only and not the color pixels.

resolution (HR) i.e. finer resolution of the patches are made available, they can be used to find a reliable match which is a better exemplar for inpainting. Considering this intuition as a cue we now discuss our proposed approach.

4.2 Proposed approach

The symbols used throughout this chapter are briefly described in table 4.1 and the steps used in our proposed approach are summarized in algorithm 1. The details of our approach are as follows. Given an image I_0 having a region Ω_0 to be inpainted, we obtain the coarser resolution image I_{-1} by blurring and downsampling I_0 as done in [51]. Let Ω_{-1} denote the missing region in I_{-1}, which corresponds to Ω_0 as shown in figure 4.3.

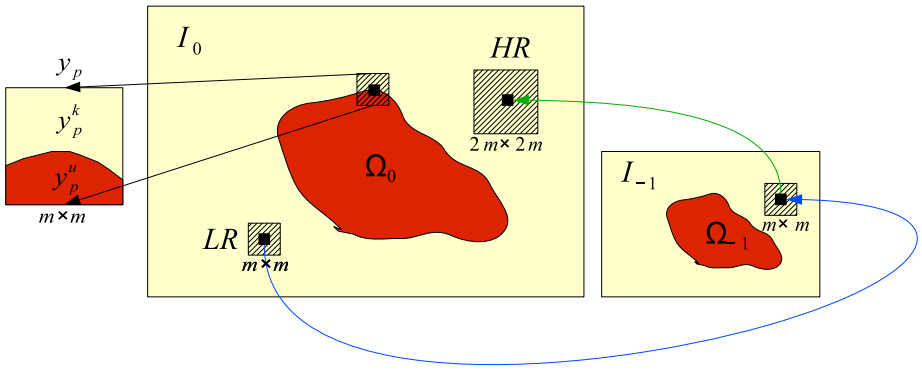


Figure 4.3: Finding LR-HR patch pairs using given image I_0 and its coarser resolution I_{-1}.

We now select a patch y_p of size $m \times m$ around a pixel p on the boundary of Ω_0 for filling, based on a priority order that depends on the presence of structure and proportion of known pixels in the patch y_p. For calculating the priority we use the method proposed in [26] that was explained earlier in section 3.2. Let y_p^k and y_p^u denote the known and the unknown pixels in y_p. The patch y_p is then compared with every $m \times m$ sized patch in the known region $I_0 - \Omega_0$ using sum of squared difference (SSD) by considering only the pixels corresponding to y_p^k. We then obtain K best matches denoted as $y_{q_1}, \ldots, y_{q_K}$ representing the candidates exemplars. The exemplar based methods [26, 149] use $K = 1$ to obtain the best match, whereas our method considers more candidate matches by setting $K > 1$

in order to find a better exemplar. These patches are then used in obtaining HR patches.

Khatri and Joshi [73] have shown that HR details can be self-learnt from the given image and its single coarser resolution. Drawing inspiration from [73], the proposed method estimates the HR details even for patches with missing pixels. To do this we first find LR-HR matches for known regions over the entire image. Consider an LR patch of size $m \times m$ in the known region $I_0 - \Omega_0$. We can obtain the corresponding $2m \times 2m$ sized HR patch in the same resolution by considering the coarser resolution I_{-1} as illustrated in figure 4.3. Although not all LR patches can find a good match in the coarser resolution, we use this methodology to create dictionaries of image-representative LR-HR patch pairs, with the help of which a

Symbols	**Meaning**
I_0, I_{-1}	Input image and its coarser resolution.
Ω_0, Ω_{-1}	Region to be inpainted in the input image I_0 and corresponding region in I_{-1}.
y_p	Patch of size $m \times m$ around a pixel $p \in I_0$.
y_p^u	Unknown missing pixels in the patch y_p that are to be inpainted i.e. $y_p^u \in \Omega_0$.
y_p^k	Known pixels in the patch y_p i.e. $y_p^k \in I_0 - \Omega_0$.
K	Number of candidate exemplars.
$y_{q_1}, \ldots, y_{q_K}$	Candidate exemplars corresponding to the patch y_p.
N	Number of patch pairs used for constructing the LR-HR dictionaries.
D_{LR}	Dictionary of low-resolution patches. Dimension: $m^2 \times N$.
D_{HR}	Dictionary of high-resolution patches. Dimension: $4m^2 \times N$.
$D_{LR_p}^k$	Dictionary of low-resolution patches containing only those rows that correspond to the known pixels y_p^k. Dimension: $\lvert y_p^k \rvert \times N$.
α	Sparse vector of size $N \times 1$.
Y_p	HR patch of size $2m \times 2m$ corresponding to LR patch y_p.
Y_p^u, Y_p^k	HR pixels in patch Y_p that correspond to the pixels y_p^u and y_p^k, respectively, in the LR patch y_p.
$Y_{q_1}, \ldots, Y_{q_K}$	HR patches corresponding to the candidate exemplars $y_{q_1}, \ldots, y_{q_K}$.
Y_q	Best match for Y_p among $Y_{q_1}, \ldots, Y_{q_K}$.
H_p	Final inpainted HR patch corresponding to the LR patch y_p.
L_p	Inpainted version of the LR patch y_p.

Table 4.1: Description of the symbols used in this chapter.

Algorithm 1 Steps used in our approach for simultaneous inpainting and super-resolution.

1: Construct LR-HR pair dictionaries using the known regions in I_0 and I_{-1}.
2: Select highest priority patch $y_p = y_p^k \cup y_p^u$ for inpainting using method in [26]. Here, $y_p^k \in I_0$ and $y_p^u \in \Omega_0$.
3: Search for candidate sources (exemplars) $y_{q_1}, \ldots, y_{q_K}$ in I_0.
4: Self-learn HR patches $Y_{q_1}, \ldots, Y_{q_K}$ and Y_p using the constructed dictionaries:
   (a) Obtain $Y_{q_1}, \ldots, Y_{q_K}$ corresponding to $y_{q_1}, \ldots, y_{q_K}$.
   (b) Estimate Y_p corresponding to y_p.
5: Find best exemplar Y_q in HR by comparing Y_p with $Y_{q_1}, \ldots, Y_{q_K}$.
6: Obtain final inpainted HR patch H_p using Y_p and Y_q.
7: Obtain inpainted LR patch L_p from H_p using transformation estimated from the constructed dictionaries and update Ω_0.
8: Repeat steps 2–7 till all patches in Ω_0 are inpainted.

good match can be estimated for any LR patch in the known region. We also learn the HR of an LR patch y_p with missing pixels (i.e. $y_p^u \in \Omega_0$) by making use of these LR-HR patch pairs. Simultaneous inpainting and SR of the missing pixels is then performed by refining the estimated HR of y_p using HR of the best candidate among $y_{q_1}, \ldots, y_{q_K}$ and an LR-HR relationship learnt from the known region. Thus, we make use of self-learning while obtaining the HR patches of inpainting region which are then used to obtain the corresponding inpainted LR patches. In what follows we provide the details of (a) constructing LR-HR patch pair dictionaries, (b) estimation of HR patches, and (c) simultaneous inpainting and SR of missing pixels.

4.2.1 Constructing LR-HR patch pair dictionaries

To obtain the image-representative LR-HR patch pairs, we consider every $m \times m$ sized patch in the known region $I_0 - \Omega_0$. For each of these patches we find the best match by searching for similar patches in $I_{-1} - \Omega_{-1}$. We then get the corresponding HR in $I_0 - \Omega_0$. Here, every LR patch will be mapped to exactly one HR patch. However, an HR patch may be mapped by many LR patches when

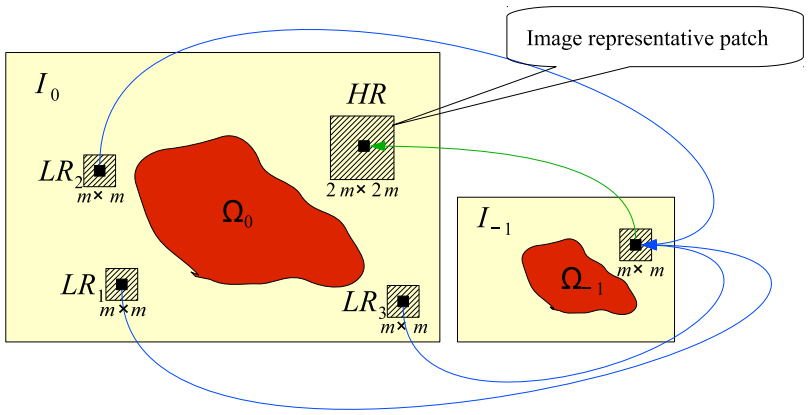


Figure 4.4: Example of an image representative patch.

the LR patches are similar as seen in figure 4.4 where LR_1–LR_3 patches map to a single HR patch in I_0. We then create a plot of HR patches versus the number of LR patches that each HR patch is mapped to. The HR patches that are highly mapped indicate repetitiveness of the LR patches and are therefore appropriate for representing the image patches. On the other hand, the HR patches having less number of LR mappings are less likely to represent the patches inside the region to be filled up. Such patches are therefore discarded.

The highly mapped HR patches form the HR dictionary D_{HR} of size $4m^2 \times N$ and the corresponding $m \times m$ sized patches in $I_{-1} - \Omega_{-1}$ form the LR dictionary D_{LR} of size $m^2 \times N$. Here N is the number of highly mapped patches such that $N >> 4m^2$. Note that the dictionaries constructed in this way do not have LR-HR pairs for every patch in the known region of I_0.

4.2.2 Estimation of HR patches

For an LR patch whose match is directly available in the LR dictionary, the corresponding patch in the HR dictionary is the required HR patch. For other LR patches we estimate a good match using a linear combination of few patches in the LR dictionary. When a signal is known to be sparse, the compressive sensing (CS) theory [17] provides a method to obtain the sparse representation. In our case, an LR patch y whose HR version needs to be estimated, can be sparsely

represented using the LR dictionary D_{LR} such that:

$$y = D_{LR} * \alpha, \tag{4.1}$$

where α is a sparse vector of size $N \times 1$ and y represents the lexicographically ordered LR patch of size $m^2 \times 1$. The sparse vector α is obtained by posing the problem as:

$$\min ||\alpha||_{l_1}, \quad \text{subject to} \quad y = D_{LR} * \alpha, \tag{4.2}$$

where $||\alpha||_{l_1}$ corresponds to $\sum_{j=1}^{N} |\alpha_j|^1$ which is minimized using standard optimization tools [16]. In this way, we obtain good matches from the already available LR dictionary itself. Assuming the LR-HR patch pairs to have the same sparseness and using the estimated sparse coefficients (α), the corresponding HR patch Y of size $4m^2 \times 1$ is obtained as follows:

$$Y = D_{HR} * \alpha, \tag{4.3}$$

where D_{HR} denotes the HR dictionary. The pixels in Y are rearranged to get a patch of size $2m \times 2m$ by reversing the operation that was used to obtain the lexicographical ordering. This procedure is used to obtain the HR patches $Y_{q_1}, \ldots, Y_{q_K}$ corresponding to the K candidate source patches $y_{q_1}, \ldots, y_{q_K}$ by replacing $y = y_{q_i}$, $\alpha = \alpha_{q_i}$ and $Y = Y_{q_i}$ for $i = 1, \ldots, K$ in the above equations (4.1)–(4.3).

The patch y_p that needs to be inpainted has missing pixels y_p^u. Therefore, one cannot directly obtain the corresponding HR patch. However, the known pixels y_p^k can be represented using a reduced LR dictionary $D_{LR_p}^k$ which consists of only those rows in D_{LR} that correspond to the pixels y_p^k depending on which of the pixels in y_p are missing. Here $D_{LR_p}^k$ is of size $|y_p^k| \times N$ where $|y_p^k|$ denotes the number of known pixels in y_p. We now obtain Y_p corresponding to y_p by replacing $y = y_p^k$, $\alpha = \alpha_p$, $D_{LR} = D_{LR_p}^k$ and $Y = Y_p$ in equations (4.1)–(4.3). Note that, in order to obtain Y_p we use the complete HR dictionary D_{HR} of size $4m^2 \times N$ and hence Y_p has the size of $2m \times 2m$, i.e. it has no missing pixels. Since Y_p is obtained

by considering only the known pixels $y_p^k \in y_p$ and the corresponding dictionary D_{LR}^k, the pixels Y_p^k that correspond to y_p^k represent true HR pixels. Likewise, the HR pixels Y_p^u that correspond to y_p^u provide a better approximation to the missing HR pixels due to the use of many similar and representative patches.

4.2.3 Simultaneous inpainting and SR of missing pixels

The final HR patch selection for missing regions is done using Y_p and $Y_{q_1}, \ldots, Y_{q_K}$ as follows. We compare each of the HR patches $Y_{q_1}, \ldots, Y_{q_K}$ with Y_p and choose the one having minimum SSD as Y_q. As the pixels in Y_p^u represent approximate but not true HR version of the missing pixels, we replace them with those in Y_q in which all pixels represent true HR. The resulting patch H_p is final HR patch which is then used to obtain the LR patch L_p representing the inpainted version of the patch y_p.

In order to obtain L_p from H_p we need the HR to LR transformation. In our case, blurring and downsampling is used to obtain coarser resolution I_{-1} from I_0 as done in [51]. Hence the same operation is used to obtain L_p from H_p. However, if the point spread function (PSF) of the camera is available, one can use it and perform downsampling to obtain the coarser resolution patches. Alternatively, if one uses I_{-1} that is captured using a camera, then the HR to LR transformation can be estimated from the available dictionaries having true LR-HR patch pairs to get L_p from H_p. Once the LR-HR dictionary pair is available we can model each LR pixel lr_i as a linear combination of 4 HR pixels $hr_i^{00}, hr_i^{01}, hr_i^{10}$, and hr_i^{11} as follows:

$$lr_i = [hr_i^{00}\ hr_i^{01}\ hr_i^{10}\ hr_i^{11}][a_{00}\ a_{01}\ a_{10}\ a_{11}]^T, \tag{4.4}$$

where a_{00}, a_{01}, a_{10} and a_{11} are the coefficients of the linear combination. Using the pixels in the LR-HR pair dictionaries in equation (4.4), these coefficients can be estimated in the least-squares sense. We can then obtain L_p from H_p by making use of the estimated coefficients.

We now have both LR and corresponding HR patches which are inpainted.

The patch H_p is now placed appropriately in the upsampled image to obtain SR of the inpainted region. This process is repeated to inpaint the entire missing region Ω_0. Note that in every iteration, only the missing pixels y_p^u in the selected patch y_p are inpainted and the missing region Ω_0 is updated accordingly. The order in which the patches are selected for filling is based on presence of structure and number of known pixels. This helps in propagating the structure inside the missing regions as a result of which the global structure is preserved. One may also super-resolve all the patches in the known region by a factor of 2 by estimating the corresponding HR patches as explained in section 4.2.2. This will result in HR image where both known and inpainted regions are super-resolved.

4.3 Experimental results

We now present the inpainted results on the natural scene dataset available in [63]. The dataset also contains results of the state-of-the-art methods for image inpainting viz. image melding [28], Photoshop CS5 content aware fill [7], statistics of patch offsets [59], GIMP Resynthesizer plugin [57], planar structure guidance [62, 149] and the method by Komodakis and Tziritas [75]. We compare the results of our proposed method with these methods. The number of candidate matches considered in our implementation is $K = 5$ and the patch size is taken to be $m = 7$. In section 3.3 we have discussed that the quality of image inpainting is best evaluated by assigning a subjective score with the help of human observers. Based on this discussion, we compared the results considering a consensus of the observations made by volunteers. We now use the same method to compare the results obtained using our proposed method with those obtained using the above mentioned state-of-the-art methods. These comparative results are presented in figures 4.5–4.10 which are discussed below.

Figure 4.5 shows the results of inpainting the marked region corresponding to one of the kids in the cage. One can see the outline of the kid as well as the rods showing inconsistent bending in the inpainted results shown in figures 4.5(c)–4.5(d). An extra arm can be seen in figure 4.5(e) while some artefacts can be seen

(a) Input (b) Region to be inpainted (c) Image melding [28] (d) Komodakis & Tziritas [75] (e) Patch offsets [59]

(f) Content aware fill [7] (g) Planar structure guided [62] (h) Planar structure unguided [62] (i) GIMP Resynthesizer [57] (j) Proposed method

Figure 4.5: Results of inpainting the marked region corresponding to one of the kids in cage.

(a) Input (b) Region to be inpainted (c) Image melding [28] (d) Komodakis & Tziritas [75] (e) Patch offsets [59]

(f) Content aware fill [7] (g) Planar structure guided [62] (h) Planar structure unguided [62] (i) GIMP Resynthesizer [57] (j) Proposed method

Figure 4.6: Results of inpainting people and vehicle near the shop

in figures 4.5(f) and 4.5(i). The results in figures 4.5(g)–4.5(h) are not only blurred, but also show inconsistency in the inpainted rods. The inpainted region obtained using the proposed method shown in figure 4.5(j) looks visually better when compared to other approaches.

The results of inpainting people and a vehicle in front of a shop are shown in figure 4.6. An implausible inpaintng of the region occluded by the vehicle can be seen in figures 4.6(c)– 4.6(d) and 4.6(f)–4.6(i). Similarly none of the results in figures 4.6(c)–4.6(i) show completion of the advertisement board occluded by the vehicle. Observe that the inpainting result of the proposed method displayed in figure 4.6(j) is not only plausible within the region but also well restores the

(a) Input (b) Region to be inpainted (c) Image melding [28] (d) Komodakis & Tziritas [75] (e) Patch offsets [59]

(f) Content aware fill [7] (g) Planar structure guided [62] (h) Planar structure unguided [62] (i) GIMP Resynthesizer [57] (j) Proposed method

Figure 4.7: Results of inpainting the table and chairs in a restaurant

(a) Input (b) Region to be inpainted (c) Image melding [28] (d) Komodakis & Tziritas [75] (e) Patch offsets [59]

(f) Content aware fill [7] (g) Planar structure guided [62] (h) Planar structure unguided [62] (i) GIMP Resynthesizer [57] (j) Proposed method

Figure 4.8: Results of inpainting benches on the hill-top.

advertisement board.

Figure 4.7 shows the inpainting of a table and chairs in a restaurant. In each of the inpainted results in figures 4.7(c)–4.7(g) a part of either chairs or table is visible, while the results in figures 4.7(h)–4.7(i) show improper inpainting of the brown tiles. The result of the proposed approach depicted in figure 4.7(j) does not show any artefacts of table or chairs in the inpainted regions and the inpainted tile region looks acceptable.

Another result in figure 4.8 shows the inpainting of benches on a hill-top. The result in figure 4.8(d) shows unrealistic criss-cross shadows of the fence, while those in figures 4.8(c), 4.8(f) and 4.8(h) have shadow of the fence in the right-half

(a) Input (b) Region to be inpainted (c) Image melding [28] (d) Komodakis & Tziritas [75] (e) Patch offsets [59]

(f) Content aware fill [7] (g) Planar structure guided [62] (h) Planar structure unguided [62] (i) GIMP Resynthesizer [57] (j) Proposed method

Figure 4.9: Results of inpainting people in front of the trucks.

of the image, which is undesirable. The result shown in figure 4.8(g) is clearly not consistent with the known regions. Similarly, figure 4.8(e) has the door extended downwards that unrealistically cuts through the floor, while figure 4.8(i) appears to have a visible seam on the boundary of the inpainted region. Note that the result of the proposed method in figure 4.8(j) does not have any unrealistic shadows and is seamlessly inpainted. The texture of the inpainted region matches well with the region surrounding it.

In order to show the effectiveness of our approach on inpainting the region with low contrast we now consider another example. These results are shown in figure 4.9. We see that in the result of the proposed method shown in figure 4.9(j), the bumper of the truck in the left side of the image is well completed on inpainting the occluding person. None of the results shown in figures 4.9(c)–4.9(h) show completion of the bumper region. Similarly, in figures 4.9(c)–4.9(i), the edge of the pavement below the bumper does not appear to be convincingly inpainted, whereas figure 4.9(j) looks better inpainted. From the all results shown in figures 4.5–4.9, it is clear that our method performs better when compared to state-of-the-

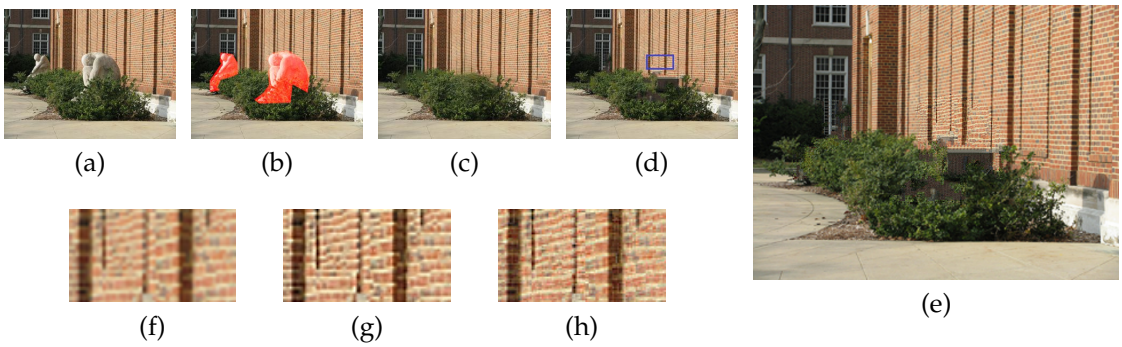


Figure 4.10: Result showing simultaneous inpainting and SR: (a) input; (b) regions to be inpainted (c) inpainting using planar structure guidance [62]; ; (d) inpainting using proposed method showing a rectangular area with blue boundary inside one of the inpainted regions; (e) simultaneously inpainted and super-resolved image (by a factor of 2) using the proposed method with known regions upsampled using bicubic interpolation; (f)–(h) zoomed versions after upsampling (the region marked by the rectangular area with blue boundary in (d)) using various approaches viz. (f) bicubic interpolation, (g) Glasner et al. 's method [51] and (h) our method for super-resolution.

art approaches. Hence one can say that use of an additional constraint of matching patches at higher resolution results in better inpainting.

In order to show the effectiveness of our approach in super-resolving in addition to inpainting, we also present a result showing SR in figure 4.10. The inpainted and super-resolved region is compared with Glasner et al. 's approach [51] where the SR is performed on our inpainted result at the original resolution. Note that SR approaches super-resolve only what is available i.e. regions having no missing pixels, whereas the missing pixels are estimated and also super-resolved in our approach. Hence, our approach not only inpaints but also reconstructs high resolution of the unknown region with missing pixels. We display the inpainted result in figure 4.10(d) and simultaneous SR in figure 4.10(e) obtained using our method. The zoomed version after upsampling one of the inpainted regions (shown by the rectangular area with a blue boundary in figure 4.10(d)) using bicubic interpolation and Glasner et al. 's method [51] for SR are depicted in figures 4.10(f) and 4.10(g), respectively. Looking at the results, we see that the super-resolved region shown in figure 4.10(h) is comparable to the SR result shown in figure 4.10(g). Also, the simultaneously super-resolved region as obtained in our approach (figure 4.10(h)) shows greater details than simply upsampling the in-

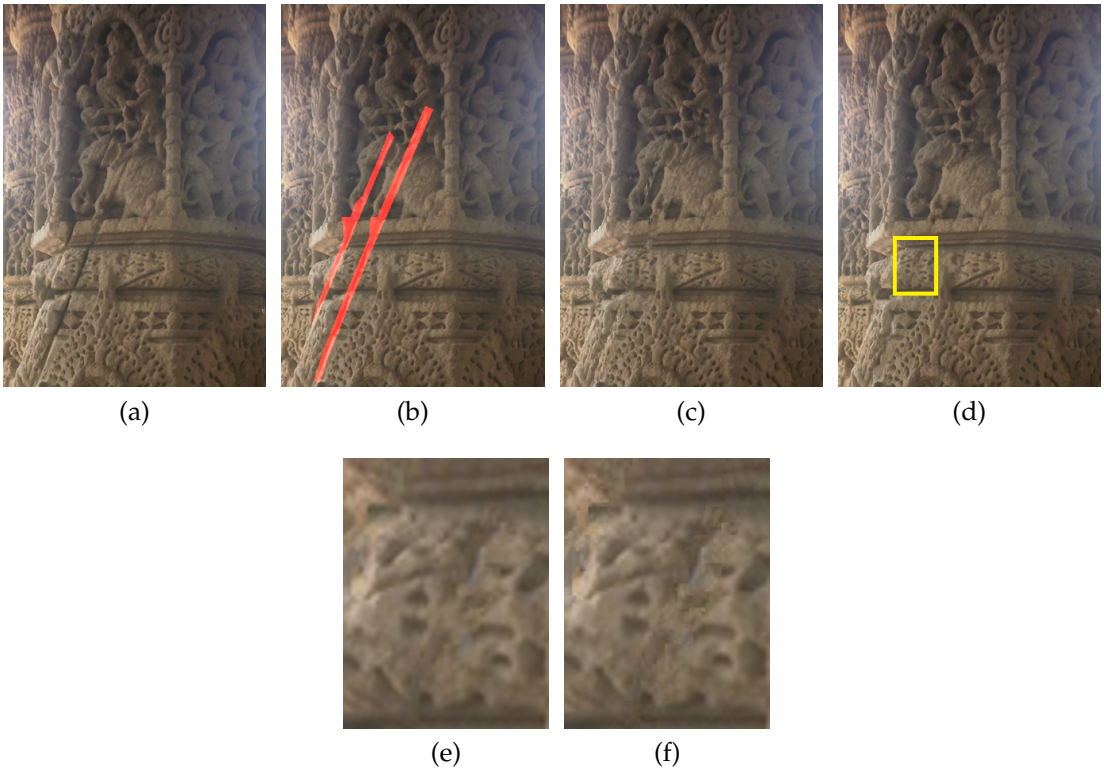


Figure 4.11: Result of simultaneous inpainting and super-resolution of the sword-marks at Sun Temple, Modhera, India. (a) Input image; (b) regions to be inpainted is shown in red color; (c) inpainted result using AR model; (d) inpainted result using proposed approach; (e) zoomed version after bicubic interpolation of the rectangular region with yellow border shown in (d); (f) zoomed version of the simultaneously inpainted and super-resolved region (by a factor of 2) corresponding to the rectangular region in (d).

painted region using bicubic interpolation as shown in figure 4.10(f).

We now show the results of proposed method in comparison with the those obtained using the AR model based inpainting technique discussed in chapter 3, on heritage site images. These results are shown in figures 4.11–4.12. Sword-marks over the historic stone carvings at the Sun temple at Modhera, India are shown in figure 4.11(b). The inpainted image using the AR model discussed in chapter 3 is shown in figure 4.11(c) while that using our proposed method is shown in figure 4.11(d). Here, one can observe that the inpainting in both the results appear to be seamless with plausible artistic work generated inside the inpainted regions. However, one may note that the result using our proposed approach shown in figure 4.11(d) looks more natural in comparison to the one shown in figure 4.11(c) when we look at the inpainted area near the elephant's trunk. Moreover, the

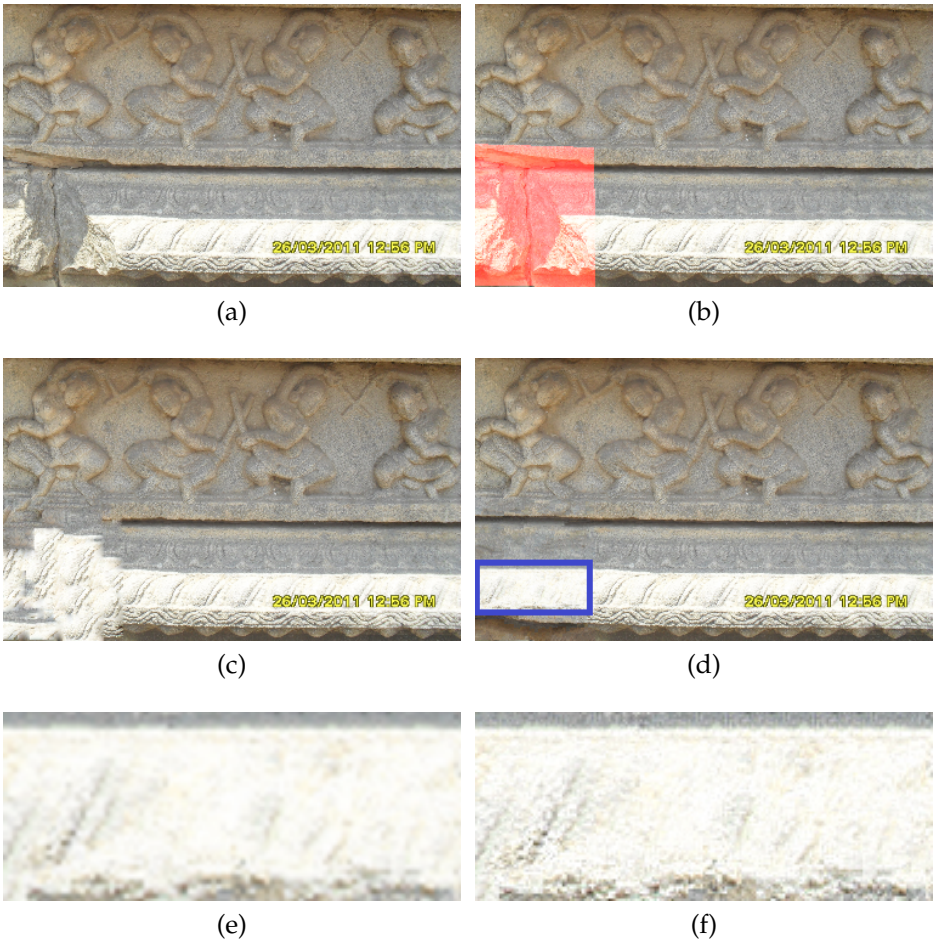


Figure 4.12: Result of simultaneous inpainting and super-resolution of a damaged wall of a temple at Hampi, India. (a) Input image; (b) region to be inpainted is shown in red color; (c) inpainted result using AR model; (d) inpainted result using proposed approach; (e) zoomed version after bicubic interpolation of the rectangular region with blue border shown in (d); (f) zoomed version of the simultaneously inpainted and super-resolved region (by a factor of 2) corresponding to the rectangular region in (d).

zoomed version of inpainted and super-resolved zoomed version shown in figure 4.11(f) provides clearer details in comparison to bicubic interpolation (figure 4.11(e)) of the inpainted region shown in figure 4.11(d).

We illustrate one more result for an input image shown in figure 4.12(a) having a damaged wall of a temple at Hampi, India as marked in figure 4.12(b). We display the inpainted image using the AR model based approach in figure 4.12(c) and that using our method is shown in figure 4.12(d). Here, one can observe the repair of a substantially large damaged region wherein the fine details

within the repaired region are also visible. One may note that, although, both models yield seamless inpainting, the result obtained using our proposed approach appears more plausible and in addition performs resolution enhancement i.e. super-resolution. This can be seen from the zoomed image shown in figure 4.12(f) which provides greater details when compared to the bicubic interpolated region shown in figure 4.12(e). Thus, in comparison to the inpainting method using the AR model discussed in chapter 3, proposed approach provides better inpainted results that look more natural and additionally also provides the resolution enhancement.

For surface reconstruction, if the point cloud data corresponding to the input images in figures 4.11(a)–4.12(a) are available, one may consider damaged regions as holes and perform repair using Poisson surface reconstruction [71]. Although this would fill the holes, it does not create artistic details inside the large missing regions. However, if the missing regions are filled in the images, and higher resolution details are also made available as done here by using our method, the resulting images can be used as source for generating 3D models of filled holes with reliable artistic details.

4.4 Conclusion

We have presented a unified approach to perform simultaneous inpainting and super-resolution. By using an additional constraint of matching patches at the original resolution as well as at the higher resolution, we not only obtain better source patches for inpainting but also have the corresponding super-resolved version. A comparison with the state-of-the-art inpainting methods shows that the inpainted results of the proposed method are indeed better. Also, the simultaneously super-resolved regions are comparable to the super-resolution of the inpainted regions obtained using the method in [51] and also show greater details than those obtained by upsampling the inpainted regions using bicubic interpolation. The simultaneously inpainted and super-resolved images can be used as source for generating 3D models with higher amount of details.

CHAPTER 5

Auto-inpainting of Damaged Regions in Facial Images of Statues

In chapters 3 and 4 we have discussed inpainting techniques wherein the regions to be inpainted are manually provided by the users. When we look at heritage monuments, especially the statues, there is a general consensus about the desire to view these without any damage to the dominant facial regions. This encourages the exploration for techniques that automatically detect the damaged regions so that their repair can be completely automated using an existing inpainting technique. Another reason that motivates this exploration is that it can be a useful tool to automate the monitoring of heritage site using a surveillance system and alert the authorities if any damage to monuments takes place. The damage could be either intentional or it could be unintentional due to the curiosity of visitors, based on which a necessary action can be initiated and monuments can be protected from further damage.

In this chapter we discuss a method that automates the process of detecting the damage to visually dominant regions viz. eyes, nose and lips in facial images of statues, and their repair using an existing inpainting approach [113]. Here, we use bilateral symmetry of face as a cue to detect the eye, nose and lip regions. Textons

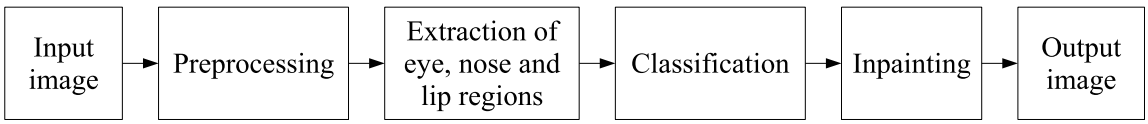


Figure 5.1: Block diagram of our approach for detecting and inpainting the damaged facial regions in statues.

features [142] are then extracted from each of these regions in a multi-resolution framework to characterize their textures. These textons are matched with those extracted from a training set consisting of true damaged and non-damaged regions in order to perform the classification. The repair of the identified damaged regions in the test image is then performed using the Poisson image editing method [113] by considering the best matching non-damaged region from the training set.

A block diagram of our method is shown in figure 5.1, the details of which are organized as follows. In section 5.1 we discuss preprocessing, followed by extraction of the eye, nose and lip regions in section 5.2. Details of classifying the detected region as either damaged or non-damaged are given in section 5.3. We discuss the repair of the detected damaged regions using inpainting in section 5.4 followed by experimental results in section 5.5 and conclusion in section 5.6.

5.1 Preprocessing

The input is assumed to be a frontal face image. One may note that slight deviation from the frontal pose is not an issue in the region detection process. However, for large deviation due to complex distortions, one may think of using image registration as a preprocessing step. Nevertheless, given a single image with complex distortions, registration itself is a difficult problem and involves pixel interpolation, thus affecting classification. We therefore consider only the frontal face images as our inputs.

The detection of the regions of interest viz. eyes, nose and lips is based on edge features and can get affected by changes in the illumination conditions. In order to make the detection process robust to illumination changes, we apply the single scale retinex (SSR) algorithm[1] [68] on the input image. Further, we apply an edge preserving smoothing operation [114] over the resulting image, so as to detect the regions with better accuracy. Following this, the edges are extracted to obtain an

[1]We used the implementation of SSR algorithm available at: http://in.mathworks.com/matlabcentral/fileexchange/26523-the-inface-toolbox-v2-0-for-illumination-invariant-face-recognition/content/INface_tool/photometric/single_scale_retinex.m [143].

edge image I_e.

5.2 Extraction of eye, nose and lip regions

The visually dominant regions viz. eyes, nose and lips have a common property of being bilaterally symmetrical. Motived by the work in [70], our approach uses this property as a cue for detecting the eye, nose and lip regions. Using the edge image I_e, symmetry measures $b_h(x,y)$ and $b_v(x,y)$ around each pixel location (x,y) are calculated in the horizontal and vertical directions, respectively as follows,

$$\begin{aligned} b_h(x,y) &= \sum_{j=1}^{\min(y,y-N)} [1(I_e(x,y-j) = I_e(x,y+j))] \text{ and} \\ b_v(x,y) &= \sum_{i=1}^{\min(x,x-M)} [1(I_e(x-i,y) = I_e(x+i,y))], \end{aligned} \tag{5.1}$$

where, $M \times N$ represents the size of input image and $1(condition)$ is an indicator function that outputs the value of 1 if *condition* is true, else outputs 0. The pixels considered for calculating $b_h(x,y)$ and $b_v(x,y)$ are illustrated in figure 5.2.

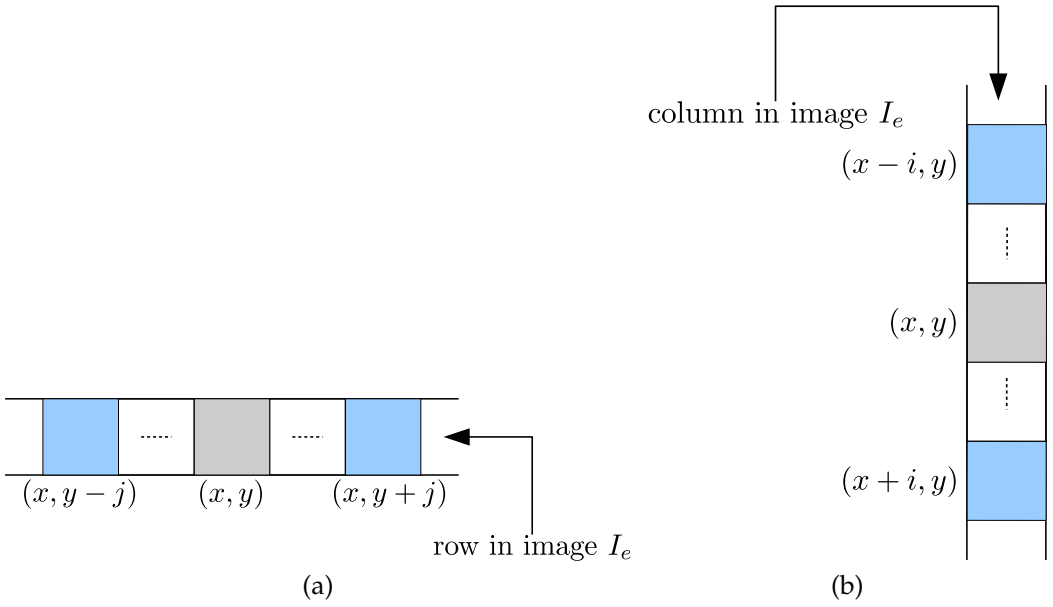
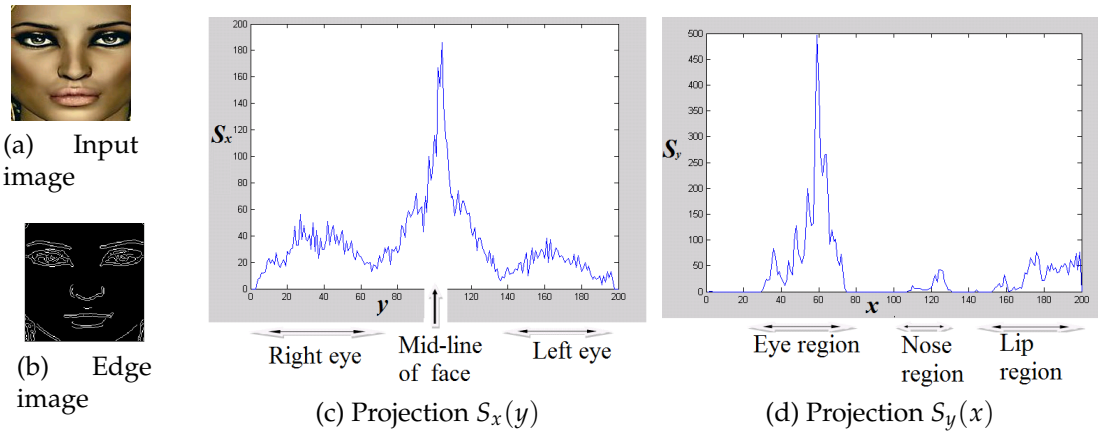


Figure 5.2: Pixels considered for the calculation of symmetry measures $b_h(x,y)$ and $b_v(x,y)$ in (a) horizontal and (b) vertical directions, respectively.

The calculated symmetry measures are then used to obtain the projections S_x



(a) Input image

(b) Edge image

(c) Projection $S_x(y)$

(d) Projection $S_y(x)$

Figure 5.3: Extraction of the potential regions of interest using bilateral symmetry.

and S_y as follows:

$$S_x(y) = \sum_{i=1}^{M} b_h(i,y), \quad \text{and} \quad S_y(x) = \sum_{j=1}^{N} b_v(x,j), \tag{5.2}$$

where, y and x respectively denote the column and row being projected. The peak in projection S_x provides the mid-line about which the face is nearly symmetric, while the peaks in projection S_y help in identifying vertical locations of the eye, nose and lip regions. This is illustrated using the example shown in figure 5.3. The regions of interest can then be extracted using appropriately sized windows around the locations of the peaks detected in projections S_x and S_y.

One may wonder if a little skew or non-perfect fronto-parallel view may jeopardize the bilateral symmetry measures. While it is true that a little skew from the non fronto-parallel position does affect the bilateral symmetry of the image, it may be noted that the peaks in the projections of the horizontal and vertical symmetry

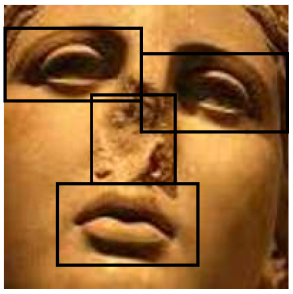


Figure 5.4: Detected eye, nose and lip regions shown in black rectangular regions for a case of little skew or non-perfect fronto-parallel image.

measures $S_y(x)$ and $S_x(y)$, respectively, get shifted in accordance to the skewness and the regions of interest get detected accordingly. An example of detecting the eye, nose and lip regions in such a case is shown in figure 5.4. However, this does not hold true for large deviation from the non fronto-parallel position, which may lead to failure.

5.3 Classification

For classifying the detected regions as damaged or non-damaged we use texture as a cue. A method for modeling different texture classes having uniformity within each class has been proposed in [142]. Our work, however, deals with the images of statues at historic monuments that have natural textures with no uniformity. In such cases, it is difficult to extract any repetitive pattern at a single scale. However, irregular patterns and structures in nature have been successfully represented using fractals [37, 84]. The fractals are geometric patterns that repeat at smaller scales to produce irregular shapes and surfaces that cannot be represented by classical geometry. This motivated us to make use of a multi-resolution framework to address the issue of irregularities in natural texture at different resolutions; a property characterized by stone-work and monument surfaces. Moreover, our method automatically calculates the number of clusters required to represent the two classes that correspond to damaged and non-damaged regions, as opposed to the approach in [142] which uses fixed number of clusters for representing several texture classes.

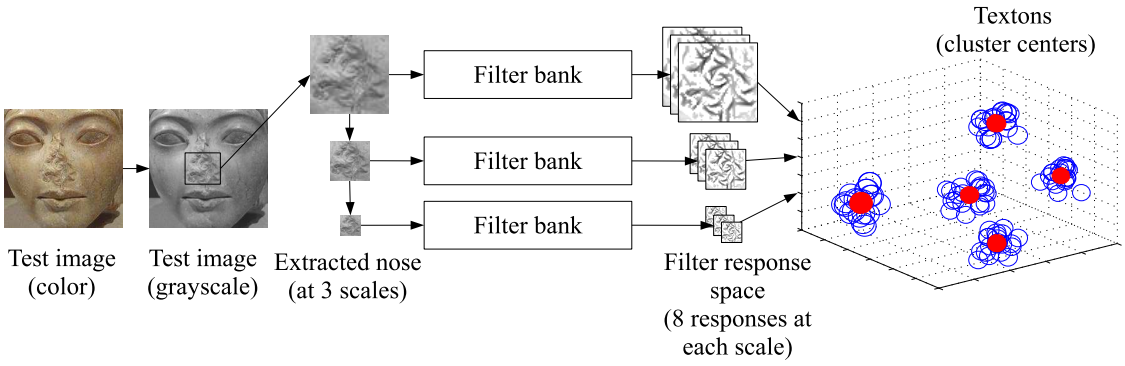


Figure 5.5: Texton extraction from the detected nose region in a test image.

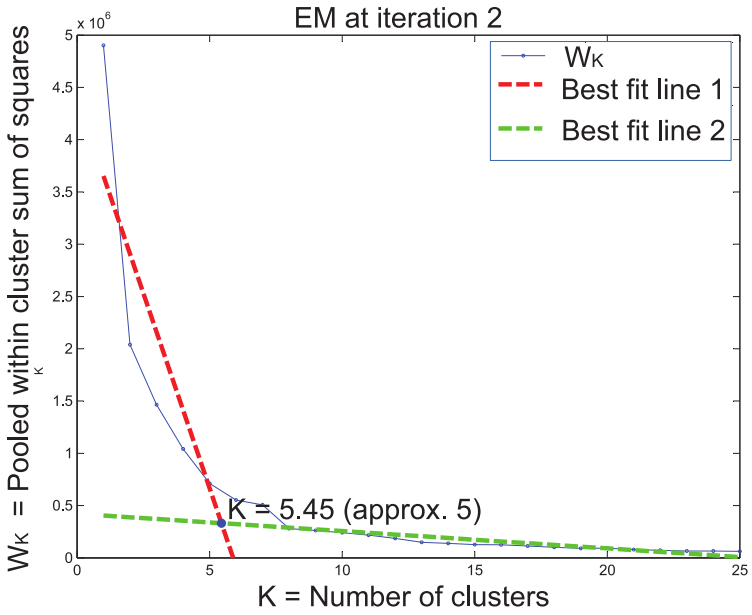


Figure 5.6: Auto-selection of number of clusters K by fitting two straight lines to the data.

The texture features are extracted in the form of textons, which are cluster centers in the filter response space. These textons are obtained in a multi-resolution framework by convolving the detected potential region of interest and its two coarser resolution versions with the maximum-response-8 (MR8) filter bank [142]. In order to obtain the coarser versions of the detected region, it is low-pass filtered using Gaussian filter before downsampling. The MR8 filter bank consists of 38 filters viz. edge and bar filters with 6 orientations at 3 scales along with a Gaussian and a Laplacian of Gaussian filter. Each pixel of the input region is now transformed into a vector of size 8 by considering 8 maximum responses out of the 38 filters. In other words, the maximum response for orientation of the edge and bar filters at each scale along with the response for the Gaussian and Laplacian of Gaussian filters are recorded to obtain a vector of size 8. We illustrate the process of extracting the textons for a detected nose region in the test image, with the help of figure 5.5. A similar process is independently applied to extract the textons features from the eye and lip regions.

The K-means algorithm is then applied on these vectors to obtain the K cluster centers i.e. textons. One may note that the method proposed in [142] requires the

number of clusters (K) to be known in advance. However, it may not be possible to pre-determine the number of clusters K as this is a data dependent term. In our work, we use a simple approach to estimate the optimal number of clusters. Here, we plot a two dimensional evaluation graph, where X-axis shows number of clusters (K) and Y-axis shows the pooled within cluster sum of squares around the cluster means (W_k) calculated as follows [135]:

$$W_K = \sum_{r=1}^{K} \left(\sum_{\forall i,i' \in C_r} d_{i,i'} \right), \tag{5.3}$$

where $d_{i,i'}$ is the squared Euclidean distance between members $(i, i^{'})$ of cluster C_r. Here, Tibshirani et al. [135] have shown that the point at which the monotonic decrease flattens markedly provides the optimal value of K. However, if the curve is smooth, it is difficult to determine where exactly this decrease flattens. We then have a challenging task to obtain the optimal value of K. To overcome this difficulty, we attempted to best fit two straight lines to the curve using expectation-maximization (EM) algorithm. The point of intersection of the two best fit lines then gives the approximate point at which the curve starts to flatten. The projected point on the axis of number of clusters is then considered as the optimal value of K as illustrated in figure 5.6.

A process as described above is used offline for extracting the textons from a training set consisting of true damaged and non-damaged regions. Here, the textons representing a damaged eye, nose or lip region are extracted using all the training images containing the corresponding true damaged region. Likewise, textons representing the non-damaged regions are extracted using the true non-damaged regions from the all training images. As discussed earlier, each texton is a vector of size 8×1 which represents the cluster center of filter responses obtained at multiple scales and a collection of these vectors forms the feature set for each region. An example showing the offline extraction textons from a training set containing damaged nose regions is illustrated in figure 5.7. We now compute the Euclidean distance between textons of the detected region (viz. eye, nose or lip) in the test image and those from the corresponding true damaged and non-damaged

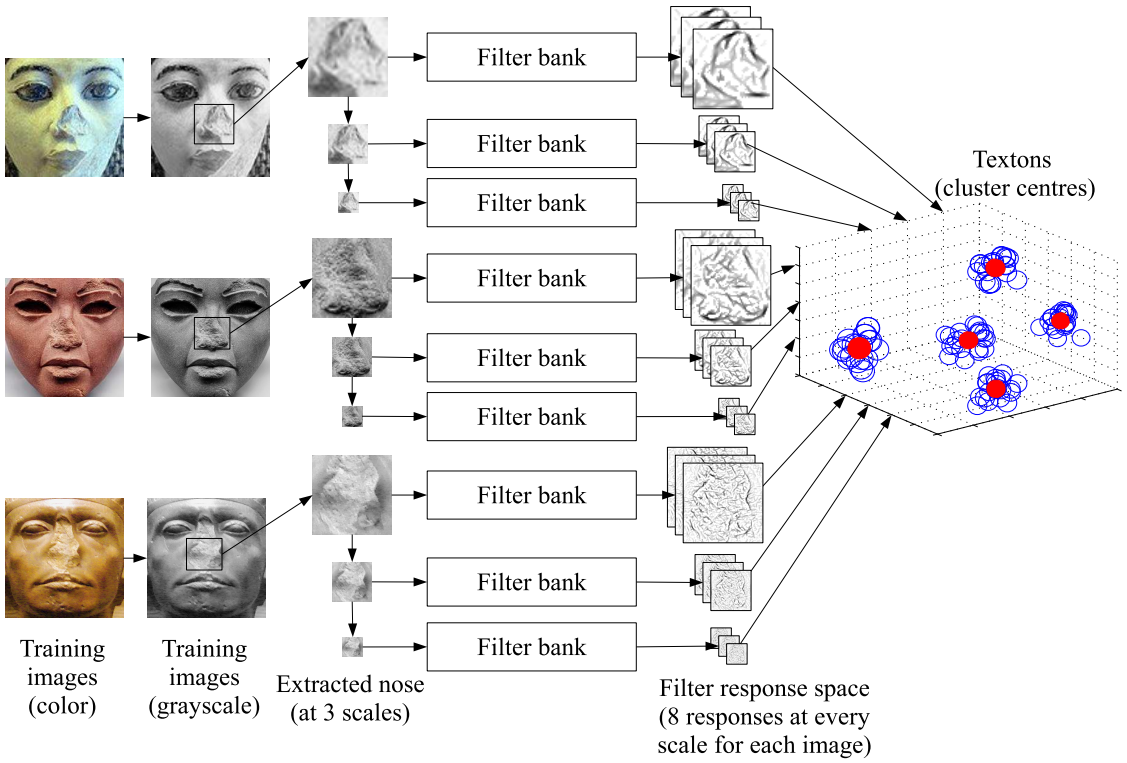


Figure 5.7: Offline extraction of textons from the training set images containing damaged nose regions.

regions of training images, to perform the classification. Here, the minimum distance criteria is used to classify the region as either damaged or non-damaged. It may be noted that for each extracted region, viz. eyes, nose and lips, the classification is performed independently. This enables the simultaneous detection of multiple damaged regions in the test image.

5.4 Inpainting

Once a region is identified as damaged, inpainting is done using a non-damaged source region. Here, if one eye is damaged, we use the flipped version of the other eye (detected automatically) from the same image as the source. However, if both eyes or the nose or lip regions are damaged, we make use of the images from the training set as the source for inpainting. Here, the source selection criteria is the extent of similarity in the Euclidean space, between the undamaged region in the test image and the training set images. However, if all the detected regions in an image are damaged, then the source regions have to be provided manually. One

Algorithm 2 Steps used in our approach for auto-inpainting (i.e. detecting and inpainting) the damaged regions in facial images of statues.

1: Make the input image illumination invariant using SSR algorithm [70] and perform edge preserving smoothing [114].
2: Extract the edges to get image I_e and calculate the symmetry measures $b_h(x, y)$ and $b_v(x, y)$ using equation (5.1).
3: Calculate the projections S_x and S_y using equation (5.2) to extract the eye, nose and lip regions.
4: Consider one detected region at a time and extract the corresponding texton features by:
   (a) obtaining the MR8 filter responses [142] using the detected region along with its two coarser resolutions and
   (b) clustering the filter responses into K clusters by auto-selecting K as shown in figure 5.6.
5: Compare the extracted textons of training and test image.
6: Identify damaged regions based on nearest neighbor criteria.
7: Repeat steps 4–6 for each detected region independently.
8: Inpaint using method in [113] by considering a suitable source region as follows:
   (a) if only one eye is damaged, use the other eye as the source.
   (b) if both eyes or other regions are damaged, use a corresponding non-damaged region from the source training set image selected based on its similarity with the non-damaged regions.
   (c) if all the detected regions are damaged, then manually provide the source regions.

way to automate inpainting in such cases is to assign a random source image. However, this may not always lead to plausible results and we therefore provide the user a choice to select the source image. Once we have the source region, we use it as a guidance vector field for the Poisson image editing technique [113] to inpaint the identified damaged region. The steps used in our proposed method are summarized in algorithm 2.

5.5 Experimental results

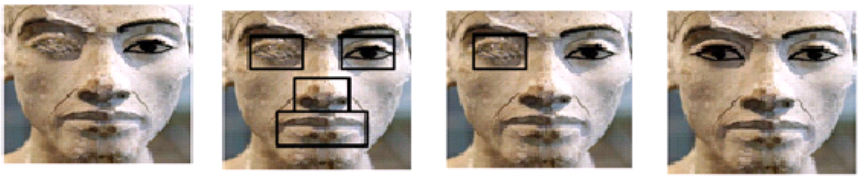
We now discuss the results of our experiments conducted on a database consisting of 40 facial images of Egyptian statues having damaged and non-damaged regions, downloaded from the Internet [53]. The spatial resolution of the images is adjusted such that all images are of the same size. A mean correction is applied to the images, so that they have the same average brightness. Training for the eye, nose and lip regions has been done independently. For training we have used 10 images each for damaged and non-damaged regions. Testing was carried out on all the images from the database including those used for training.

The results using our approach are shown in figures 5.8–5.11. The detection and inpainting of a damaged nose is shown in figure 5.8, where the source used for inpainting is an image from the training set containing an undamaged nose. In figure 5.9, the reflected version of the non-damaged left eye has been used to inpaint damaged right eye. However, in figure 5.10 since both eyes are damaged, an image from the training set containing non-damaged eyes is used as the source for inpainting. Note that the criteria used for selecting the source is similarity of the non-damaged regions in the test image with the corresponding regions in the images from the training set. In figure 5.11, we show a result where our method fails to detect the damaged nose. Here, the input image contains the nose region having small amount of damage, due to which the corresponding textons match those of the non-damaged nose regions from the training set. This is caused by the extracted statistics of the damaged and non-damaged regions. Thus, among the extracted potential regions of interest shown in figure 5.11(b), the damaged nose is incorrectly classified as undamaged and is therefore undetected in figure 5.11(c).

We now discuss the performance of our method of automatic detection of facial regions and inpainting by considering the ground truth from the inputs provided by the volunteers. Performance evaluation is done in terms of the standard recall and precision metrics defined as: Recall$= \frac{|Ref \bigcap Dect|}{|Ref|}$ and Precision$= \frac{|Ref \bigcap Dect|}{|Dect|}$. Here, *Ref* are the regions declared to be damaged or undamaged by volunteers

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

Figure 5.8: Detecting and inpainting a damaged nose; (a) input image, (b) extracted potential regions of interest, (c) detected damaged nose, (d) inpainted nose using the source image (e).



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

Figure 5.9: Detecting and inpainting a damaged eye; (a) input image, (b) extracted potential regions of interest, (c) detected damaged eye, (d) inpainted eye.

and *Dect* are the regions detected as damaged or undamaged by the proposed technique. From a set of 40 images, 50 regions were found to be damaged, while 50 were undamaged. Out of 50 damaged regions 47 were correctly classified, while all 50 undamaged regions were correctly classified. For source region selection, 49 out of 50 regions were correctly selected. The performance in terms of the recall and precision metrics is summarized in Table 5.1 which shows the effectiveness of our method.

Note that the source selection method used in our approach is not comparable with content based image retrieval (CBIR) techniques. This is because for a large damaged region, a CBIR system may not find adequate amount of non-damaged content to retrieve a good match relevant for inpainting. Although our method is developed for images of statues, it can be effective for facial regions in natural images as both have same the facial characteristics. Thus, we have presented

Region type	**#Regions**	**Recall**	**Precision**
Damaged	50	0.9400	1.0000
Undamaged	50	1.0000	1.0000

Table 5.1: Performance in terms of recall and precision.

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

Figure 5.10: Detecting and inpainting damaged eyes; (a) input image, (b) extracted potential regions of interest, (c) detected damaged eyes (d) inpainted eyes using the source image (e).

(a) (b) (c)

Figure 5.11: Failure case; (a) input image, (b) extracted potential regions of interest, (c) damaged nose is incorrectly classified as undamaged.

a texture based approach to automatically detect the damaged regions in facial images of statues for performing their digital repair using an existing inpainting technique. The results show that these regions can be effectively repaired.

5.6 Conclusion

In this chapter we have presented techniques for automatic detection of damaged dominant facial regions in statues and cracks in heritage monuments for their digital repair. In our first approach, a bilateral symmetry based method is used to identify the eyes, nose and lips. Here, texton features are extracted from each of these regions in a multi-resolution framework to characterize the textures of damaged and non-damaged regions. These textons are matched with those extracted from a training set of true damaged and non-damaged regions for detecting the damaged ones which are then inpainted with the help of suitable source regions. Here, we have addressed the repair of specific regions viz. the facial regions of statues in heritage monuments. However, damage like cracks in the non-facial regions of the monuments also diminishes their attractivenesses. We address this

in the following chapters 6–7 wherein we present techniques for automating the digital repair of cracks.

CHAPTER 6

Pixel and Patch based Approaches for Auto-inpainting of Cracked Regions

In chapter 5, we discussed a technique for automatic detection and inpainting of the damaged eye, nose and lip regions in facial image of statues. In this chapter, we consider the damage in non-facial regions and describe techniques for their automatic detection and inpainting. In particular we consider the non-facial region that have damage in the form of cracks. In heritage sites, the cracks could be developed over a period of time due to environmental effects or due to manual destruction. The presence of cracked regions make the heritage monuments less attractive. It would be interesting for the visitors if they are presented with a view of these moments after a seamless removal of the cracks. A technique for automatic detection of the cracks in an acquired image will be particularly useful for performing on-the-fly digital reconstruction when the tourists take pictures of the heritage monuments using their handheld image / video capturing devices. Towards this end, in this chapter we discuss techniques for automatic detection of cracked regions and demonstrate their repair using existing inpainting algorithms. Note that the application is to actually restore a heritage scene, i.e., digitally repair cracks that physical objects have. Thus, we are not talking about image restoration, but about object completion. In other words, we do not detect an external damage or defect due to alteration of a photograph, but instead detect and inpaint the cracked regions in the photographed scenes / objects.

The contents of this chapter are organized as follows. In section 6.1 we discuss a simple yet effective pixel based method to automatically detect the dam-

aged regions which are characterized by abruptly dark deteriorations in the photographed monuments of a heritage site. The use of order-statistics and density filters makes our technique computationally fast and we are able to accurately detect these cracked regions. In section 6.2 we discuss another approach for crack detection. This approach is based on comparing overlapping adjacent patches using singular value decomposition (SVD). The conclusion of this chapter is given in section 6.3.

6.1 A Simple Pixel based Method

Images of heritage sites usually contain murals and monuments having a porous surface, making the detection of damaged region exhibiting visual discontinuities, a very challenging task. Here, we discuss a computationally efficient and effective technique to detect these visual discontinuities which appear as abruptly dark regions representing the cracks. A block diagram of our proposed method for crack detection is shown in figure 6.1 and the details of each step are discussed in sections 6.1.1–6.1.4 followed by the experimental results in section 6.1.6. We proceed with the discussion of this proposed approach.

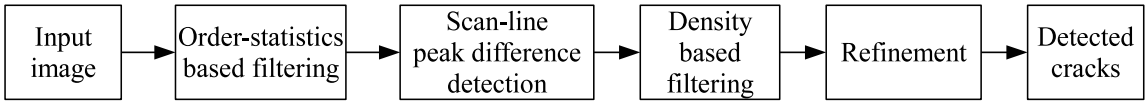


Figure 6.1: A simple approach for crack detection.

Given an input image I of a damaged monument in the RGB color space, we transform it to the HSV color space and extract the grayscale image I_V that corresponds to the *intensity* image. In the images of heritage monuments, the damaged regions like cracks usually appear darker in comparison to their surroundings area. However, due to the porous surface of photographed monuments, one observes large variations in intensity of many adjacent pixels. It is therefore necessary to enhance the contrast of these dark regions with respect to the surrounding regions. A widely used method for contrast enhancement is the histogram equalization. This is followed by edge detection to detect intensity contrast among

neighboring pixels [52]. Our experiments, however, revealed that contrast enhancement using histogram equalization increased the contrast of various other regions as well, and subsequent edge detection resulted in many false positives. We therefore, instead, use order-statistics based filtering as discussed below.

6.1.1 Order-statistics based filtering

Order-statistics filters have been used for edge enhancement in noisy images [82, 83]. This motivated us to use order-statistics filters instead of histogram equalization for enhancing the contrast of dark regions. There are various filters based on order-statistics such as the *min*, *max* and *median* filters. We use a combination of the *min* and *max* filters to achieve the desired contrast enhancement.

Since the *min* and *max* filters enhance the dark and bright regions in the image, by taking an average of the output of these filters we obtain a smoothed version I_a of the intensity image I_V. This helps us to overcomes the rapid variation in intensity due to porous regions. The *max* filtered version of I_a is then subtracted from the intensity image to achieve the contrast enhancement of the abruptly dark regions. We achieve this by considering a patch Φ_p around every pixel p in the image I_V and obtain the *min* and *max* filtered images by extracting the minimum and maximum intensities from each patch Φ_p. The average of these filters i.e. I_a is obtained using the equation (6.1) as follows.

$$I_a(p) = \frac{min(\Phi_p) + max(\Phi_p)}{2}, \quad \Phi_p \in I_V. \tag{6.1}$$

The contrast enhanced image I_h is then obtained as follows using equation (6.2),

$$I_h(p) = I_V(p) - max(\Phi_p), \quad \Phi_p \in I_a. \tag{6.2}$$

The input image I and the contrast enhanced image I_h are shown in figures 6.2(a) and 6.2(c), respectively. One may note that the contrast enhanced image I_h is not the same as the intensity image with inverted thresholds. Due to the porous surface of the photographed monuments, the intensity image I_V (shown in figure

(a) Input image I (b) Intensity image I_V (c) Contrast enhanced image I_h

Figure 6.2: Contrast enhancement using order-statistics based filtering.

6.2(b)) and its thresholds inverted version contain random sharp variations in contrast. Whereas, the image obtained after contrast enhancement as shown in figure 6.2(c) exhibits sharp contrast variations at the regions with abrupt changes in the intensity, with smooth contrast variations elsewhere. Thus, we observe a drastic variation in contrast of the cracks with respect to their surrounding regions.

6.1.2 Scan-line peak difference detection

Since the contrast enhanced image I_h exhibits sharp variations at the potentially damaged regions, we detect these by comparing adjacent pixel values along a scan-line. Large peaks of a such comparison between adjacent pixel values of a scan-line correspond to the potential cracks on that scan-line. In order to detect these cracks, we consider every scan-line to contain at most α number of peaks to be a part of the cracked regions. Here, we use a set M_x for denoting these α number of peaks per scan-line as follows,

$$\begin{aligned} M_x = \quad & \{\lambda_j, j = 1, \ldots, \alpha | \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n, \\ & \lambda_k = d_k(x, y), k = 1, \ldots, n\}, \quad \text{where,} \end{aligned} \tag{6.3}$$

$$d(x, y) = \quad \max(|I_h(x, y) - I_h(x, y + i)|), i = \pm 1,$$

where $\lambda_1, \lambda_2, \ldots, \lambda_n$ are the pixel value differences between the adjacent columns y at every scan-line x, arranged in descending order. The parameter α controls the number of high peak-differences i.e. pixels in the potential cracked regions, that can be detected along a scan-line. Using the set M_x per scan-line we create a

binary image I_b denoting the candidate pixels in the potential cracks as,

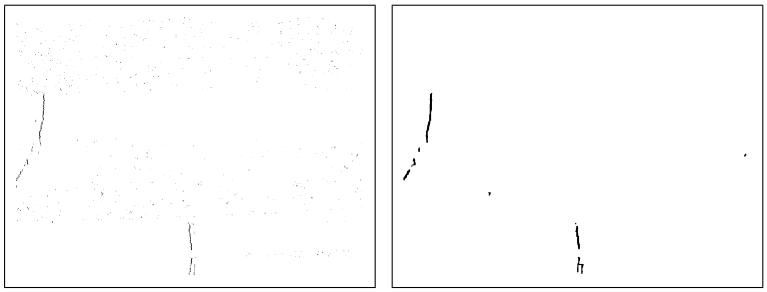
$$I_b(x,y) = \begin{cases} 1, & \text{if} \quad d(x,y) \in M_x \\ 0, & \text{otherwise.} \end{cases} \tag{6.4}$$

6.1.3 Density based filtering

Using such a process for peak-difference detection along a scan-line detects pixels at nearby locations in adjacent scan-lines and leads to increase in their density along the potential cracks. We now use a density based filter to threshold a local region φ_p around every pixel p detected in the binary image I_b based on presence of other detected pixels inside φ_p. This discards the isolated pixels detected in I_b which do not belong to the potential cracks. Thus, we localize the cracks by creating another binary image I_d as follows,

$$I_d(\varphi_p) = \begin{cases} 1, & \text{if} \quad \sum\limits_{q \in \varphi_p} \frac{I_b(q)}{|\varphi_p|} \geq \theta_1 \\ 0, & \text{otherwise,} \end{cases} \tag{6.5}$$

where $|\varphi_p|$ is the area of the local region φ_p around a pixel p and θ_1 is the threshold density. The binary image I_b and the density filtered binary image I_d are shown in figure 6.3.



(a) (b)

Figure 6.3: Detection of candidate pixels in the potential cracked regions. (a) The binary image I_b generated using equation (6.4) by detecting peak differences along every scan-line the contrast enhanced image I_h; (b) the density filtered binary image I_d.

6.1.4 Refinement

The density filtered binary image I_d, however, consists of disjoint parts of the cracks. In order to address this issue, we use the morphological dilation process so that the disjoint regions get connected. By doing so, few isolated group of pixels that do not belong to the cracked regions may also get connected. These are eliminated by applying yet another density based filtering operation on the centroids of each connected components in I_d to get a the final binary image I_g containing the detected cracks. To do this, let us denote the dilated version of I_d by I_D and a patch around centroid C_i of every connected region $i \in I_D$ by $\widehat{\varphi}_i$. The density based filtering operation on I_D is then performed using equation (6.6) as follows,

$$I_g(\widehat{\varphi}_i) = \begin{cases} I_D(\widehat{\varphi}_i), & \text{if} \quad \sum_{q \in \widehat{\varphi}_i} \frac{I_D(q)}{|\widehat{\varphi}_i|} \geq \theta_2 \\ 0, & \text{otherwise.} \end{cases} \tag{6.6}$$

Here, I_g is the output of the density filtering operation on I_D, $|\widehat{\varphi}_i|$ is the area of the patch $\widehat{\varphi}_i$ and θ_2 is the threshold.

6.1.5 Inpainting

We illustrate the regions detected in the binary image I_g in red color by overlaying them on the input image I as shown in figure 6.4(a). The regions detected in I_g represent the cracks and can therefore be used as a mask indicating the missing

(a) Input image I overlaid with the crack detected in I_g shown in red color

(b) Inpainted version of I using the region detected in I_g

Figure 6.4: Automatic detection and repair of cracks.

regions for inpainting. One can now use a suitable inpainting algorithm to perform their digital repair. We demonstrate the repair of the detected cracks using our inpainting technique discussed in chapter 3. The inpainted version of the input image I by considering the regions detected in the binary image I_g as missing regions, is shown in figure 6.4(b).

One may note that the objective of this method is to detect abruptly dark regions that characterize cracks and is therefore not suited for detection of vandalized eye, nose and lip regions of faces in statues as discussed in chapter 5 for which texture is used as a cue. Hence, the technique discussed in chapter 5 is not suited for addressing the detection of cracks.

6.1.6 Experimental results

We present the results of our technique on the data collected from the world heritage site at Hampi, India. The images were captured using a Samsung ES55 digital camera. The data consists of a number images of monuments, having both cracked and non-cracked regions. Fairly large cracks are visible in all the images. The images used in the experiments are of size 684×912 pixels. For the order-

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

Figure 6.5: Results: (a) input images; (b) cracks marked by the volunteers are shown in red color; (c) inpainted version of (b); (d) automatically detected cracks using our method are shown in red color; (e) inpainted version of (d).

statistics based filtering operation we considered patches Φ of size 3×3. For detecting the peak difference in the scan-line of the contrast enhanced image I_V, we have set the value of α to 2. The patch size φ for density based filtering operation is taken to be 9×9 while for refinement $\widehat{\varphi}$ is chosen as 61×61. The thresholds θ_1 and θ_2 are set to 0.08 and 0.45, respectively.

Since no ground truth data is available that presents the true pixels in the cracked region, we have manually determined them with the help of volunteers. Although the subjectively generated ground truth data may not be accurate, it can still be used to judge the effectiveness of the detected cracks. The results of this approach in comparison to the manually marked cracks by volunteers are shown in figure 6.5. Here, figure 6.5(a) shows the input images. The cracks marked manually by volunteers are shown in figure 6.5(b), while figure 6.5(c) shows the corresponding inpainted images. The cracks detected automatically using our technique are shown in figure 6.5(d), while figure 6.5(e) shows their inpainted versions. The results reported in figure 6.5 show that automatically detected cracked regions cover almost all the pixels marked as cracks by the volunteers. Use of these regions as input mask for inpainting techniques is justified from the inpainted results.

We now quantify the accuracy of the detected cracks using the standard recall and precision metrics defined in equation (6.7) as follows,

$$Recall = \frac{|Ref \bigcap Dect|}{|Ref|} \qquad \text{and} \qquad Precision = \frac{|Ref \bigcap Dect|}{|Dect|}, \tag{6.7}$$

where *Ref* are the pixels declared to be in the cracked regions by volunteers and *Dect* are the pixels detected as cracks by our technique. For detected cracks in

Images	**# Cracked pixels**	**Recall**	**Precision**	**Time (sec)**
Image in row 1	6501	0.9812	0.3254	3.3852
Image in row 2	3529	0.9717	0.2986	3.4164
Image in row 3	5500	0.9955	0.3082	3.3540

Table 6.1: Performance evaluation for the automatically detected cracks shown in figure 6.5(d).

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

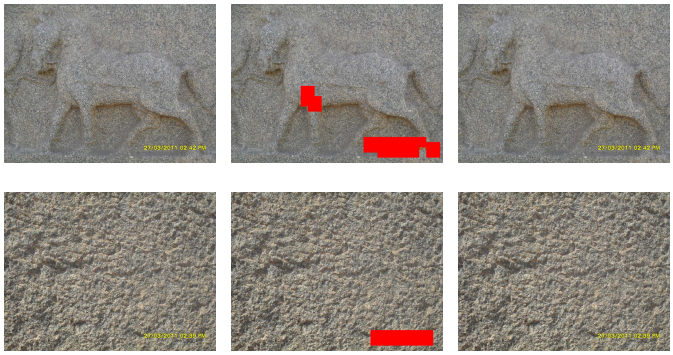
Figure 6.6: Comparative results with respect to the technique proposed in [4]. (a) Input images; (b) regions marked as cracks by volunteers; (c) detection using the technique in [4]; (d) detection using our method; (e) inpainted version of (d).

the binary image I_g to be suitable for use in an inpainting algorithm, the desired *Recall* value should be nearer to 1. A lower value indicates that less number of pixels in the cracked regions have been detected due to which the undetected cracked pixels are used as source for inpainting leading to poor inpainted results. On the other hand a low *Precision* value can be acceptable as it indicates that more number of pixels have been detected which only increases the size of the region to be inpainted. The *Recall* and *Precision* values for the results shown in figure 6.5(d) along with the time taken for detection are given in table 6.1. Here, we observe that the detected cracks have a high *Recall* value, indicating that almost all the cracked pixels marked by the volunteers are detected.

We now compare our results with those obtained from the defect detection method proposed by Amano [4] as shown figure 6.6. It may be noted that the

Images	**# Cracked pixels**	**Defect detection method [4]**			**Proposed method**		
		Recall	**Precision**	**Time (sec)**	**Recall**	**Precision**	**Time (sec)**
Image in row 1	3494	0.0000	0.0000	109.00	0.9825	0.1763	3.5256
Image in row 2	2831	0.3727	0.0239	096.00	0.9414	0.1284	3.3384

Table 6.2: Performance comparison with respect to the method in [4] for the results shown in figure 6.6.



(a) (b) (c)

Figure 6.7: Results for images that do not contain cracks. (a) Input images; (b) detection result of the technique proposed in [4]; (c) detection results using our technique.

results for technique [4] are obtained after fine-tuning the parameters. Here, we observe that the defect detection method [4] detects superimposed text but fails to detect the cracks. On the other hand, the regions detected by our method are similar to the regions marked as cracks by the volunteers. This is also suggested by the *Recall* and *Precision* values given in table 6.2. Moreover, the timing information suggests that the detection of cracks using our method is significantly fast. Here, both the techniques have been implemented on the same machine having an Intel Core i5 processor.

We now present the detection results for images that do not contain the cracked regions in figure 6.7. Here, we again observe that the defect detection technique in [4] detects the superimposed text. On the other hand, our method does not detect any region thus avoiding false detection. In other words, our method is also capable of correctly identifying cases in which no cracks are present.

We now analyze the computational complexity of our proposed technique. Consider the grayscale intensity image I_V to be of size $M \times N$, having $\eta = MN$ number of pixels, while the patch Φ having κ number of pixels such that $\kappa << \eta$. The number of pixels processed by each of order-statistic filters viz. *min* and *max* is $\eta\kappa$. Also $2\eta\kappa$ number of pixels are processed when calculating I_a which is the average of outputs of the order-statistic filters. Likewise, the calculation of the contrast enhanced image I_h involves the processing of $2\eta\kappa$ number of pixels. Further, the generation of the binary map I_b requires the processing of 2η number

of pixels. Let k denote the total count of pixels processed in all further operations, which is the product of number of pixels detected in the binary map I_b and the size of neighborhood for each operation. However, both these quantities are very less as compared to η and therefore their product $k << \eta$. The total number of pixels processed by our technique is therefore $T = 5\eta\kappa + 2\eta + k$. If processing each pixel takes a unit time, then total time required by our technique is $5\eta\kappa + 2\eta + k = O(\eta)$, which is linearly proportional to the size of the grayscale intensity image I_V.

From the results shown in figures 6.5–6.7, along with the performance comparison shown in tables 6.1 and 6.2, it is clear that the cracked regions are successfully detected by the our method. Although the technique in [4] is good for detection for an alteration to the photograph (like overlay text), our method is more suitable when it comes to detection of damage in the photographed scene / object and is comparatively faster. For successful commercialization of the auto-detection of cracks, one needs to obtain the results in quick time. Our method is straightforward and the speed of our approach is linearly proportional to the size of the input image making it usable in real-time applications. This makes our algorithm suitable for the use in digital cameras to obtain on-the-fly automatic detection of the cracks and their inpainting when a tourist is capturing the photograph of a damaged heritage scene.

We now proceed with the discussion of different approach for detection of cracked regions in section 6.2, which is patch based unlike the pixel based approach presented here.

6.2 A Patch based Approach using SVD

In section 6.1, we discussed a simple yet effective technique for detecting and inpainting the cracked regions in heritage monuments. The method is intuitive and relies on the detection of drastic variations in the intensity of adjacent pixels across a scan-line to detect the boundary of the cracked regions. Yet, it is a primitive approach that performs pixel based comparisons and involves many

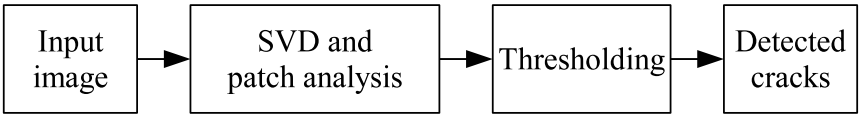


Figure 6.8: SVD based approach for crack detection.

parameters that are set heuristically.

In this section, we discuss a patch based method for detecting the cracks. Here, the main idea is to compare the overlapping patches for similarity using the singular value decomposition (SVD) based patch analysis. Calculating an average dissimilarity of the row and column adjacent patches with respect to a patch under consideration helps to reveal the amount of visual discontinuity between the patches. By using a threshold, the cracks can then be identified as the ones having higher dissimilarity value. Once the cracks are detected, their filling can be performed using a generic inpainting technique. For illustration purpose, we demonstrate this repair using the existing inpainting technique proposed in [26].

A block diagram of the proposed approach is shown in figure 6.8. Given an input image I in the RGB color space, we first transform it into HSV color space and extract the grayscale image I_V that corresponds to the *intensity* image. Now consider a patch Φ_p of size $m \times n$ at pixel $p \in I_V$ with coordinates (x, y) as shown in figure 6.9. Here, $x = 1, \ldots, M - m$ and $y = 1, \ldots, N - n$, such that $M \times N$ rep-

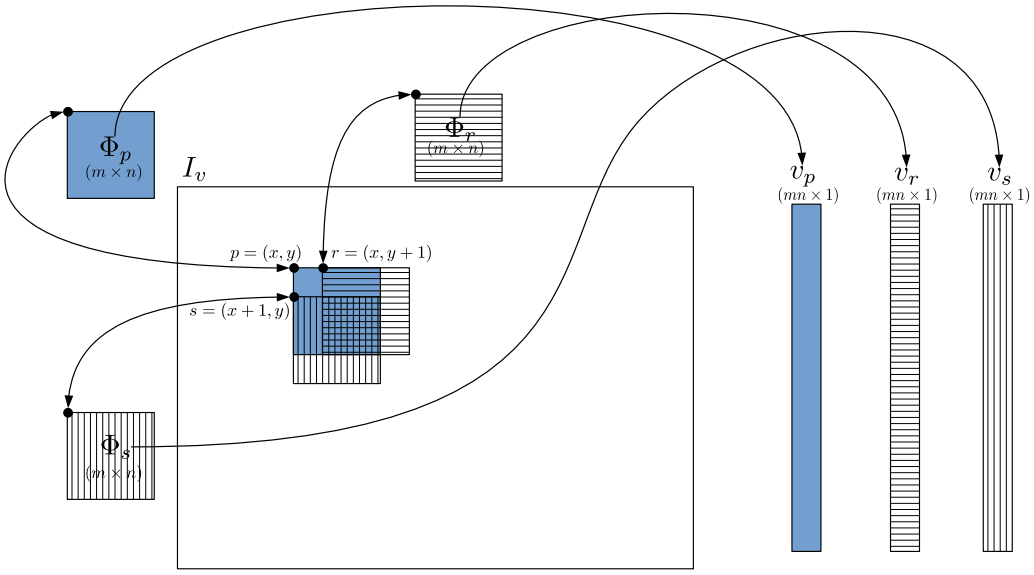


Figure 6.9: Overlapping patches considered for comparison.

resents the size of the image I_V. The elements of patch Φ_p are rearranged to form a column vector v_p of length $L = mn$ by using lexicographical ordering of pixels. Similarly, consider the pixels r and s adjacent to p with respective coordinates as $(x, y+1)$ and $(x+1, y)$. The corresponding patches Φ_r and Φ_s are also ordered lexicographically to obtain the vectors v_r and v_s as illustrated in figure 6.9.

We find the similarity between the vectors v_p, v_r and v_s using the geometric interpretation of the SVD model [30] on the matrix having these vectors as its columns. By calculating the similarity between vectors of adjacent patches, we create a similarity matrix S whose elements are then compared with an automatically estimated threshold δ, to detect patches having discontinuities. In what follows, we discuss patch analysis in the SVD domain.

6.2.1 SVD and patch analysis

We form a matrix A with the columns as v_p, v_r and v_s corresponding to patches Φ_p, Φ_r and Φ_s, and decompose it using SVD such that $A = U\Sigma V^T$. Here U is a $L \times L$ matrix, the columns of which are the eigenvectors of AA^T, V is a 3×3 matrix consisting of eigenvectors of A^TA, and Σ is $L \times 3$ matrix of singular values ($\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0$) at diagonals. We now reduce the size of matrices U to $L \times 3$ and Σ to 3×3, which however does not affect the reconstruction of $A = U\Sigma V^T$. By discarding the smallest eigenvalue, we further reduce the size of matrices U to $L \times 2$, Σ to 2×2 and V to 3×2, which now leads to an approximate reconstruction of matrix A. Such a method is widely used for image compression and noise reduction [145].

Considering the updated matrices Σ and V, the rows w_1, w_2 and w_3 of the matrix $V\Sigma$ now reflect the extent to which pixels in the corresponding columns v_p, v_r and v_s have a similar pattern of occurrence [30]. The extent of similarity between any two columns of the matrix A is given by the cosine of angle between corresponding rows of the matrix $V\Sigma$ as follows,

$$cos(\theta_{pr}) = \frac{w_1.w_2}{||w_1||\,||w_2||} \quad \text{and} \quad cos(\theta_{ps}) = \frac{w_1.w_3}{||w_1||\,||w_3||}. \tag{6.8}$$

Note that if the smallest eigenvalue is retained then the complete reconstruction of A is possible, and in that case the angle obtained by directly considering the columns of matrix A is the same as that obtained between the corresponding rows of matrix $V\Sigma$. Here, however, discarding the eigenvectors corresponding to the smallest eigenvalue helps in calculating the true similarity even when the patches are noisy.

One may easily verify this from the following example. Consider the vectors $v_p = [4,5,6,7,6,5,4]^T$, $v_r = [4,4,6,8,6,4,4]^T$ and $v_s = [1,2,3,4,3,2,1]^T$. In SVD domain representation of a matrix having these vectors as its columns, the rows of matrix $V\Sigma$ are obtained to be $w_1 = [-14.21, 0.89]$, $w_2 = [-14.12, -0.41]$ and $w_3 = [-6.52, -1.05]$, while $cos(\theta_{pr}) = 0.9958$ and $cos(\theta_{ps}) = 0.9756$. On other hand, if we directly use the vectors v_p, v_r, v_s instead of w_1, w_2, w_3, respectively, we get $cos(\theta_{pr}) = 0.9926$ and $cos(\theta_{ps}) = 0.9734$. Thus, it may be noted that unlike calculating the correlation directly between the actual patches, our method performs the similarity comparison in the SVD domain wherein by discarding the smallest eigenvalue and the associated eigenvector, the obtained similarity values are robust to noisy patches.

The comparison of patches Φ_r and Φ_s, which overlap with and are row, column adjacent to the patch Φ_p, enables us to simultaneously capture horizontal, vertical and diagonal discontinuities. We now create a similarity matrix S such that its element $S(p)$ represent the average similarity value of patch Φ_p with overlapping patches Φ_r and Φ_s calculated as follows,

$$S(p) = \frac{1}{2}\left[cos(\theta_{pr}) + cos(\theta_{ps})\right], \quad \forall p \equiv (x,y) \in I_V. \tag{6.9}$$

6.2.2 Thresholding

Once the similarity matrix S is obtained, the idea is to detect the cracks by thresholding values of the matrix S which may have different values for different input images. It may be noted that if the overlapping patches in an input image have very high similarity, then the corresponding matrix S may have many values that are nearer to 1, and therefore a high value of threshold δ could be required for

correctly detecting the patches having discontinuities that represent the cracks. Therefore, one has to choose the threshold δ based on the input image for correct detection of cracked regions, which we describe as follows.

In order to select the threshold value δ dynamically for a given image, we consider three quantities derived from the similarity matrix S, viz. the average value $avg(S)$, minimum value $min(S)$ and the maximum value $max(S)$. Since the compared patches are adjacent and also overlap each other, they have a high amount of similarity in their content. Therefore, it is reasonable to assume that the values in matrix S that are less than the average value $avg(S)$ would definitely correspond to the patches having discontinuities and hence represent cracks. Thus, the lowest value that δ may take is $avg(S)$.

If the difference between lowest and highest values of S is high, it suggests that the values corresponding to patches with discontinuities are spread over a wider range, whereas the spread is over a narrow range when the difference is small. If the values in the similarity matrix S vary in a narrow range, then the threshold value δ would be nearer to $avg(S)$. Thus, we infer that the threshold value has to be higher than the average value $avg(S)$ and also should depend on the minimum $min(S)$ and maximum $max(S)$ values. We set an initial threshold α to be an average of these three terms as given in the following equation (6.10).

$$\alpha = \frac{min(S) + max(S) + avg(S)}{3} \tag{6.10}$$

However, experimentally we found that a correction factor depending on the value α is required for correct detection of the cracks. Based on our experimentation, we arrive at the following equation (6.11) that incorporates suitable correction factors to determine the threshold δ.

$$\delta = \begin{cases} \alpha + 0.10, & \text{if} \quad 0 \leq \alpha < 0.90, \\ \alpha + 0.05, & \text{if} \quad 0.90 \leq \alpha < 0.95, \\ \alpha + 0.01, & \text{if} \quad 0.95 \leq \alpha < 0.99, \\ \alpha, & \text{if} \quad \alpha \geq 0.99 \end{cases} \tag{6.11}$$

In this way, the initial threshold α is calculated automatically, based on which an appropriate correction factor is added, to dynamically set the threshold δ depending on the input image.

Here, since overlapping patches are expected to have very high similarity values, in equation (6.11) the correction factors are added to have the threshold δ to be as near as possible to "1". Thus, for lower values of the initial threshold (α), high correction factors are added while for higher values of α low correction factors are added. While this results in a saw-tooth type variation, it helps to set thresholds nearer to "1" for α values that are on the higher side of the three ranges viz. $[0, 0.90)$, $[0.90, 0.95)$ and $[0.95, 0.99)$.

Once we obtain the threshold δ, we use it to detect the cracks by thresholding the values in the matrix S representing the similarity of patches in the intensity image I_V. If $S(p) < \delta$, we declare the corresponding patches Φ_p, Φ_r and Φ_s to be significantly dissimilar. Using this criteria, all the values in the matrix S are compared with threshold δ to detect dissimilar patches, using which a binary image B is constructed as,

$$\begin{aligned} B(\Phi_p) &= \begin{cases} 1, \text{ if } S(p) \geq \delta, \forall p \equiv (x, y) \in I_V, \\ 0, \text{ otherwise,} \quad \text{and} \end{cases} \\ B(\Phi_s) &= B(\Phi_r) = B(\Phi_p). \end{aligned} \tag{6.12}$$

The binary image B generated in this way by thresholding has the cracked regions represented by the value 1.

6.2.3 Inpainting

Once the cracks are detected, we use an existing inpainting technique to inpaint them as done in section 6.1.5. The detected cracked pixels considered as missing pixels for this purpose. Now, any generic inpainting technique can be used to perform inpainting of the missing pixels, and to show this, here we use the inpainted technique proposed in [26], which is a different inpainting technique than the one used in section 6.1.5. In the following section 6.2.4, we illustrate the detection or cracks and their inpainting using few experimental results, in order to show the

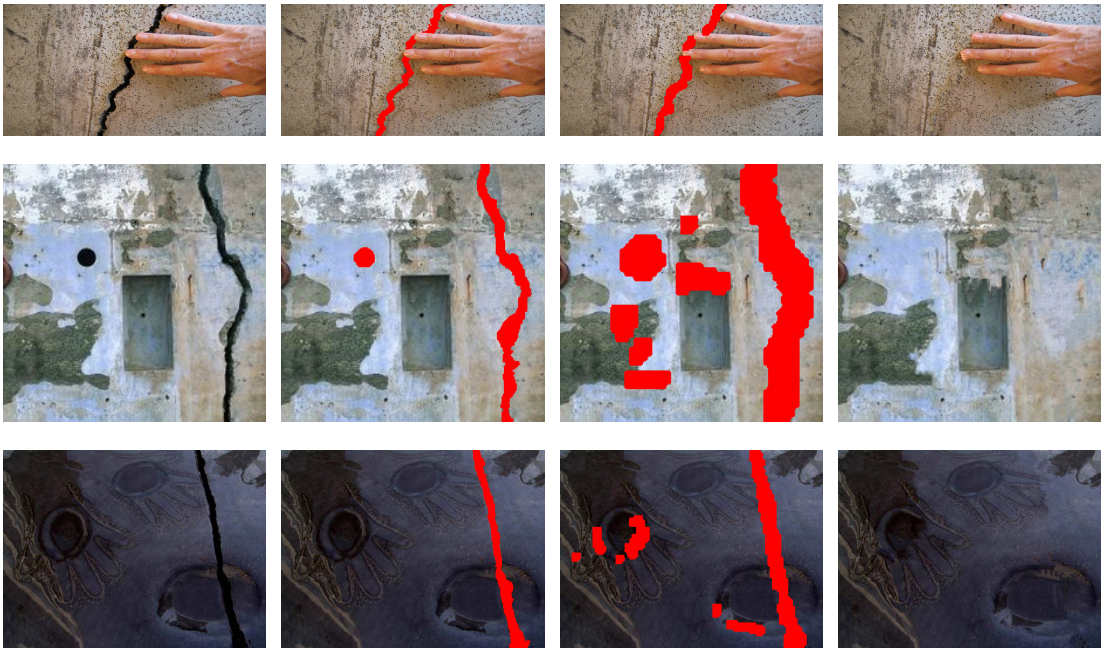
working of the proposed SVD based approach.

6.2.4 Experimental results

We now discuss the results of our experiments on images downloaded from the Internet [53], as well as on those captured by us. In our experiments, the size patches Φ_p, Φ_r and Φ_s is set to 3×3. As mentioned earlier in section 6.2.2, the detected cracks are inpainted using the technique proposed in [26] in order to demonstrate the suitability of our method to auto-detect cracked regions for inpainting.

The results of our experiments on wall and ceiling images are shown in figure 6.10, and those on images pavements are illustrated in figure 6.11. The input images of wall and ceiling are shown in figure 6.10(a). Since the ground truth containing the marked cracked regions was not available for the input images, we have considered regions marked by volunteers as cracks for comparison. The images containing the regions marked as cracks by volunteers are shown in figure 6.10(b). The detected cracks using our method are shown in figure 6.10(c) and the corresponding inpainted images are shown in figure 6.10(d). Here, the input image in the first row contains a crack having high contrast with respect to its surroundings. Although this appears to be a trivial case, one may note that the presence of tiny dark regions over the image make the detection a challenging task. The detection performed by our method is similar to the region marked by the volunteers. The input image in the second row contains cracks along with other damaged regions. In this case also our method well detects the cracked region. A more complex case containing the crack in an image having low contrast is shown in the input image in the third row. Even in this case our method performs better. Likewise, in the pavement images shown in figure 6.11, the detected cracked regions are similar to those marked by the volunteers, indicating the efficacy of our crack detection method.

In order to quantify the accuracy of the detected cracks, once again we consider the popularly used recall and precision metrics [160] defined in equation (6.7). The performance in terms of *Recall* and *Precision* values for input images in figures



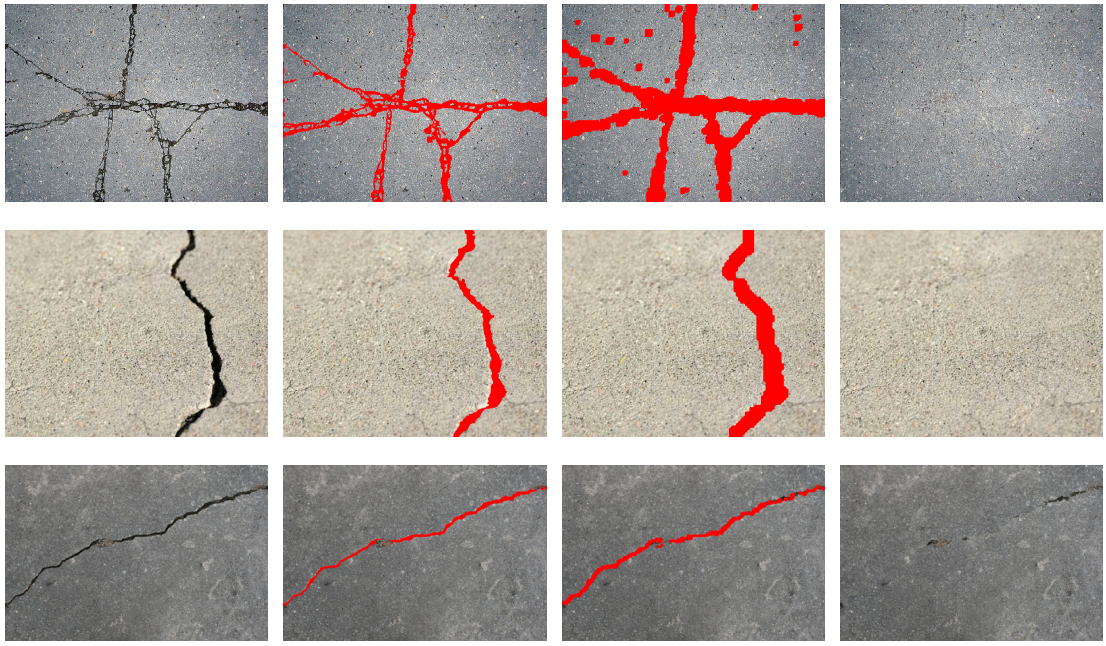
(a) Input (b) Cracks marked by volunteers (c) Detected cracks (d) Inpainted image

Figure 6.10: Detection of cracks in images of wall and ceiling. (a) Input images; (b) Cracks marked by volunteers are shown in red color; (c) Cracks detected using our method; (d) Inpainting of the detected cracks in (c) using the technique proposed in [26].

6.10 and 6.11 is given in table 6.3. The crack detection results obtained using our technique are significantly similar to the detection performed manually by volunteers which is also evident from table 6.3. We observe that *Recall* value for all the detected cracks in these images is nearer to 1. This clearly indicates that the desired pixels of the cracked regions have been detected.

Input	**# Cracked Pixels**	**Recall**	**Precision**
Image in row 1 of figure 6.10	05414	0.9540	0.6702
Image in row 2 of figure 6.10	02513	0.9988	0.2290
Image in row 3 of figure 6.10	05431	0.9742	0.3708
Image in row 1 of figure 6.11	40741	0.9914	0.4445
Image in row 2 of figure 6.11	05613	0.9984	0.4772
Image in row 3 of figure 6.11	29333	0.8919	0.5781

Table 6.3: Performance of the proposed technique in terms of Recall and Precision for the results shown in figures 6.10 and 6.11.



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

Figure 6.11: Detection of cracks in pavement images. (a) Input images; (b) Cracks marked by volunteers are shown in red color; (c) Cracks detected using our method; (d) Inpainting of the detected cracks in (c) using the technique proposed in [26].

We now show results on images captured by our camera in figure 6.12, that include indoor and heritage scenes. These results are compared with the defect detection technique proposed in [4]. It may be noted that the results for technique [4] are obtained after fine-tuning the parameters while the parameters α and δ in

Input	**# Cracked Pixels**	**Defect detection method [4]**			**Proposed method**		
		Recall	**Precision**	**Time (sec)**	**Recall**	**Precision**	**Time (sec)**
Image in row 1	8217	0.1503	0.0093	512	1.0000	0.5372	4.63
Image in row 2	1353	0.6438	0.0260	093	1.0000	0.1749	4.41
Image in row 3	3494	0.0000	0.0000	109	0.9531	0.2971	4.51

Table 6.4: Performance comparison in terms of Recall and Precision for the images shown in figure 6.12.

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

Figure 6.12: Detection of cracks in indoor and heritage scene images captured by us. (a) Input images; (b) Cracks detected using the technique proposed in [4] are shown in red color; (c) Cracks detected using our method are shown in red color. (d) Inpainted image for the cracks detected by our method.

our method are dynamically calculated, depending on the input image. The performance comparison for these results in term of recall and precision are shown in table 6.4. From the results shown in figure 6.12 and the performance comparison in table 6.4, it is clear that the desired cracked regions are successfully detected by our method. Although the technique in [4] is good for detection for an alteration to the photograph like overlay text, our method is comparatively fast and more suitable when it comes to detection of damage in the photographed scene / object.

In comparison to our previous crack detection approach discussed in section 6.1, the recall using the proposed approach is more or less similar but the precision shows improvement indicating higher accuracy in detecting the cracks. This is also indicated by one of the results in tables 6.2 and 6.4 corresponding to, (a) an image of a heritage site shown in row 1 of figure 6.6 ($Recall = 0.9825$ and $Precision = 0.1763$), and (b) the same image shown in row 3 of figure 6.12 ($Recall = 0.9531$ and $Precision = 0.2971$).

6.3 Conclusion

In this chapter we have first presented a simple yet effective crack detection technique that can be used to generate an input mask for inpainting algorithms, considering the case of inpainting images of heritage sites. Our proposed method automatically detects the cracks region which are characterized by abruptly dark deteriorations. By performing scan-line peak difference detection on contrast enhanced potential cracked regions, followed by a density filtering operation, we localize the cracked regions. The refinement of this output provides the area that contains the cracked regions which can be repaired by using a generic inpainting algorithm. This method is fast, which makes it suitable for the use in real-time applications. We have then presented a patch based technique for detection and inpainting of cracks by comparing similarity of overlapping patches in the SVD domain. Here, we have used an image adaptive threshold to detect the dissimilar patches that indicate visual discontinuities and construct a binary map containing the cracked regions. Experimental results show that the proposed method performs better crack detection in comparison to the technique in [4] and is more accurate than the premitive pixel based crack detection method discussed in section 6.1. Nevertheless, the detected regions using both the proposed techniques can be suitably used as input masks to inpaint the images containing cracked regions.

In the following chapter 7, we propose yet another but more accurate approach to automatically detect and inpaint cracks, along with its extension to perform auto-inpainting in videos of heritage sites containing cracked regions.

CHAPTER 7

Tolerant Edit Distance based Auto-inpainting of Cracked Regions

In chapter 6 we have discussed pixel and patch based crack detection methods for inpainting. In this chapter, we present another effective and more accurate crack detection technique based on patch comparison using a measure derived from edit distance [144] used for comparing text strings. We elaborate details of the proposed crack detection technique in section 7.1 for images. We then consider the extension of this technique in section 7.2 for performing auto-inpainting of video frames by making use of the scale invariant feature transform (SIFT) and homography. We consider the camera movement to be unconstrained while capturing video of the heritage site, as such videos are typically captured by novices, hobbyists and tourists. Here, we also provide a novel measure to quantify the quality of inpainted videos. The conclusion of this chapter is given in section 7.3.

7.1 Crack Detection using Tolerant Edit Distance

Cracks are typically characterized by dark areas in an image which can be easily identified by human beings but pose difficulty to computers. In trivial cases, simple thresholding is sufficient for detecting the cracks. However, as seen previously in chapter 6, the subtle variations in pixel intensities make the detection of cracks a challenging task. The approach that we now discuss uses similarity of non-overlapping adjacent patches as a cue to localize the cracked regions.

A block diagram of this approach for crack detection is shown in figure 7.1.

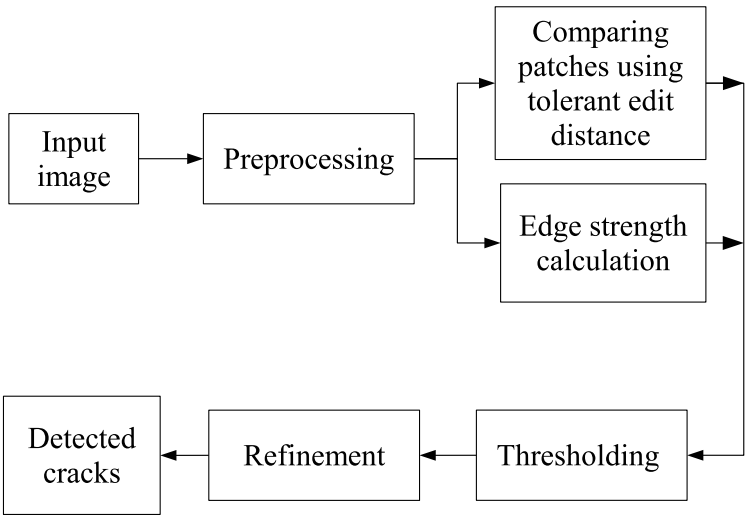
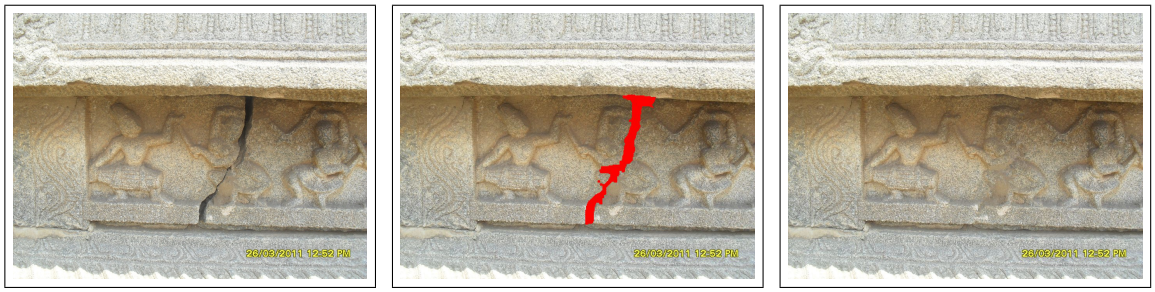


Figure 7.1: Our approach for crack detection using tolerant edit distance.

The novelty here is that we compare the patches for similarity using a measure derived from the edit distance, which is a popular string metric used in the area of text mining. Our distance measure is such that it avoids penalizing trifle differences between the corresponding pixels of the compared patches. This helps in better localization of the cracks since the patch similarity is determined by avoiding a strict pixel-to-pixel comparison inside the patches. The patch penalty along with average edge strength within the patches is used to localize the cracked regions. Further, the refinement of the localized cracked regions is performed using a sophisticated segmentation technique unlike our previous patch based approach, which did not perform refinement. The overall process in the proposed approach thus provides the cracked regions detected in more a accurate manner.



(a) (b) (c)

Figure 7.2: Auto-inpainting cracked regions. (a) Original image of a heritage scene. (b) Automatically detected cracked region using the proposed method is shown in red color. (c) Image obtained after inpainting the detected region.

We then demonstrate the digital repair of the detected cracked regions using an existing inpainting technique [26]. One such example is illustrated in figure 7.2.

We now provide the details of our proposed approach, starting with the preprocessing step.

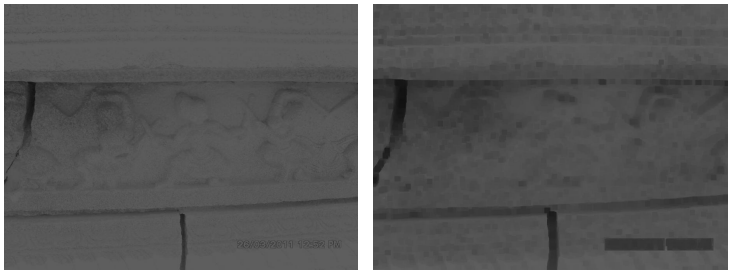
7.1.1 Preprocessing

For a given input image I of size $M \times N$ we perform a preprocessing step by considering its intensity normalized version I_0. Since the cracked regions are dark, the low intensity pixels are more likely to be part of a crack. We construct a weight matrix I_w from I_0 such that dark pixels have higher weights given by,

$$I_w(x, y) = \exp(-I_0(x, y)), \tag{7.1}$$

where (x, y) denote the pixel coordinates. The weights in I_w are multiplied with the corresponding pixels in I_0 and the resulting image is eroded to obtain an image I_v that we use for further processing. The erosion operation is performed so that the narrow dark regions are enlarged for proper detection, which may otherwise

(a) I (b) I_0 (c) I_w



(d) $I_0 * I_w$ (e) I_v

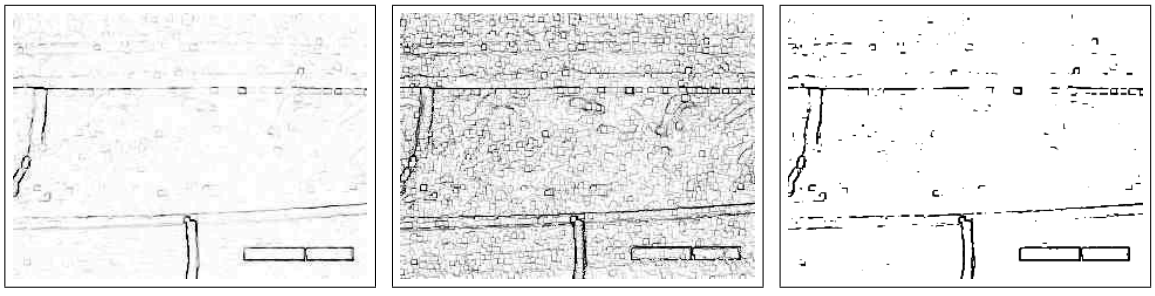
Figure 7.3: Preprocessing of an input image.

remain undetected in later operations. The preprocessing step is illustrated with an example in figure 7.3.

7.1.2 Patch comparison using tolerant edit distance

Since the cracked regions exhibit noticeable dissimilarity with respect to the neighboring regions, we intend to mark them out by comparing adjacent non-overlapping patches in the image I_v. A simple method for patch comparison is to calculate sum of absolute difference or sum of squared difference (SSD) across corresponding pixels of the compared patches. These measures are, however, sensitive to noise and may give a high error even for visually similar patches, which is evident in figure 7.4. Moreover, comparing a patch with its spatially shifted version also gives high error, where in fact both are visually identical. Thus, it becomes difficult to separate the cracks from its neighborhood using a threshold.

In string matching, shifting errors are overcome using the edit distance [144]. Edit distance is a string metric that gives the count of operations required for transforming one string into another. The transformation is achieved by comparing the characters of first string with that of the second string and performing an appropriate operation. Here, the valid operations on comparing a pair of characters are insertion, deletion and substitution. For example, consider two strings "books" and "loops". Here only two operations, both substitutions viz. "b" to "l" and "k" to "p" are required for the transformation. Hence the edit distance



(a) (b) (c)

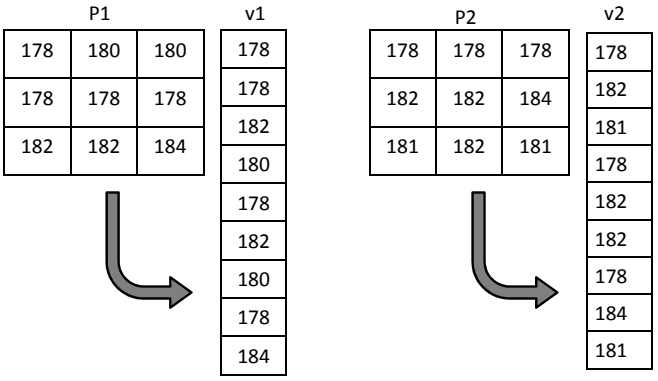
Figure 7.4: Comparison of (a) sum of absolute difference image, (b) sum of squared difference image and (c) tolerant edit distance image I_{tED} for tolerance $\delta_t = 10$. Patch size is 3×3. With the input image of size 684×912 we have I_{tED} of size 227×303. Here, an enlarged, intensity inverted version is shown for clarity.

between "books" and "loops" is 2. Likewise, for transforming "books" to "oops" we again require two operations, a deletion and a substitution, giving an edit distance equal to 2.

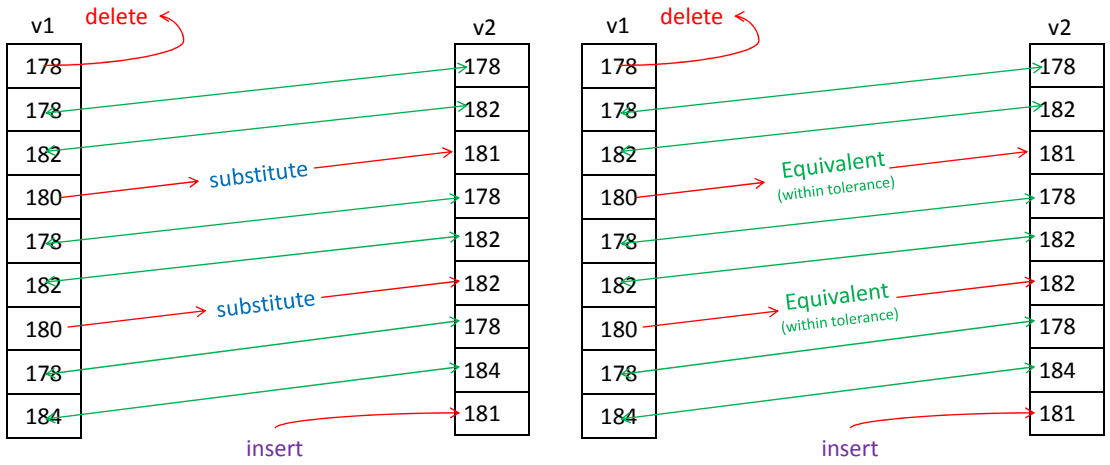
Now, in order to compare patches, consider the lexicographical ordering of two patches synonymous to two strings and each pixel synonymous to characters of the respective strings. If we calculate the edit distance, it would give the number of operations required to transform one patch to another. A smaller value of edit distance conveys less number of operations and in turn higher similarity of the patches. However, in the presence of noise, the edit distance will still be higher. This is because the substitution operation penalizes the mismatch of compared characters.

In order to overcome the noise sensitivity of edit distance, a tolerance can be used for the substitution operation. In other words, if the difference between the compared characters falls within some tolerance value, the characters can be considered as equivalent and, therefore, no penalty is given by the substitution operation. We call the edit distance with such a substitution operation as tolerant edit distance (tED). The tED thus gives a measure of similarity between patches, in the presence of noise and spatial shift. To illustrate this, consider the example shown in figure 7.5 where we compare two visually similar patches P1 and P2 of size 3×3, such that the patch P2 is a vertically shifted version of patch P1. Here, the value for edit distance is 4 due to the shifting. However, by considering pixel values within a tolerance = 3 as equivalent, we have the tolerant edit distance = 2 indicating robustness to noisy pixel values. One may note that if these patches are compared using sum of absolute difference or SSD we get higher comparison error. We therefore use tolerant edit distance to compare patches of size $m \times n$ in I_v after the preprocessing step. This is done as follows.

For a patch Φ_p at pixel p with coordinates (x, y) in the image I_v, the right and bottom non-overlapping patches that are considered for similarity are patches Φ_r and Φ_s at pixels $r = (x, y + n)$ and $s = (x + m, y)$ as shown in the figure 7.6. One may also consider a diagonal neighboring patch at pixel $(x + m, y + n)$. Nevertheless, the right and bottom neighboring patches suffice to localize that cracked



(a) Lexicographical ordering of patches P1 and P2 to v1 and v2



Edit distance = 4
(1 delete + 2 substitute + 1 insert = 4 operations)

(b) Edit distnace (i.e. tolerance = 0) between v1 and v2

tolerant Edit distance (for tolerance = 3) **= 2**
(1 delete + 0 substitute + 1 insert = 2 operations)

(c) Tolerant edit distnace for tolerance = 3 between v1 and v2

Figure 7.5: Example for comparing two patches of size 3×3 using edit distance and tolerant edit distance. Here, the patch P2 is a vertically shifted version of patch P1.

pixels and the accuracy of detecting the cracked regions is improved by refinement. Let the pixels in patch Φ_p, and its right and bottom neighboring patches Φ_r and Φ_s be rearranged using lexicographical ordering to form vectors v_p, v_r and v_s, respectively.

We now measure the similarity between patches Φ_p, Φ_r and Φ_s by calculating the tolerant edit distance (tED) d_{pr} and d_{ps} between the pairs v_p, v_r and v_p, v_s, the average of which is assigned to the pixel p. The algorithm for computing the tED is given in algorithm 3. The tED calculated using algorithm 3 is such that pixel values within a tolerance δ_t are considered to be equivalent. It is calculated for all

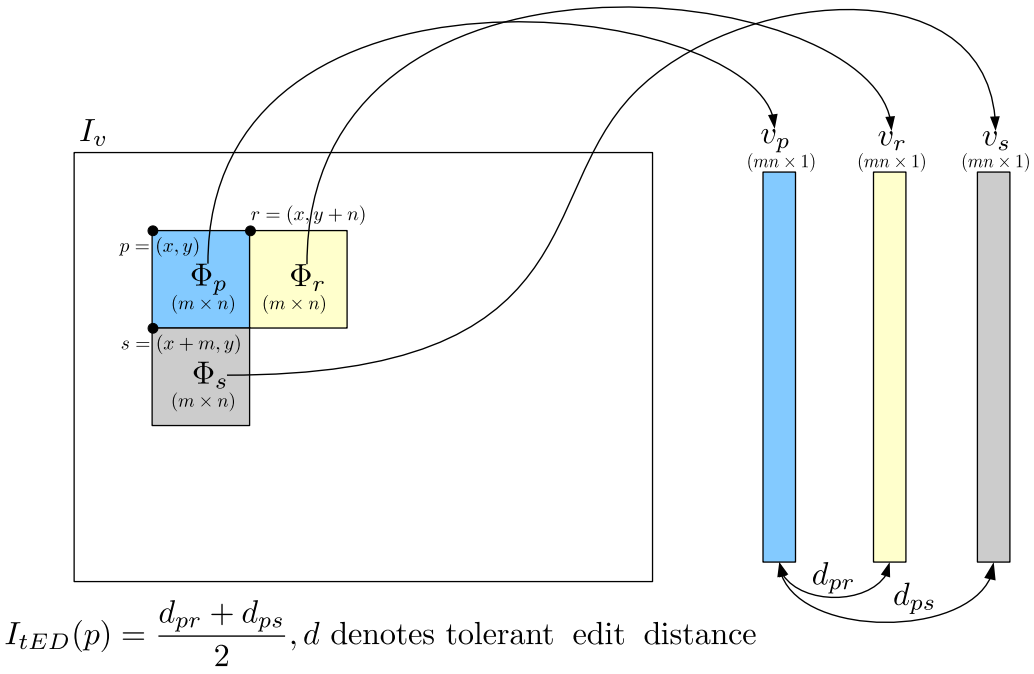


Figure 7.6: Patch comparison using tolerant edit distance.

the patches for which there exist both right and bottom non-overlapping adjacent patches. The calculated tED values are used to form an image I_{tED}, which, when multiplied with an edge strength image makes it easier to detect the cracked regions. Figure 7.4(c) shows the image I_{tED} corresponding to image I_v depicted in figure 7.3(e).

7.1.3 Edge strength calculation

Since the cracked regions are distinct from their neighboring regions, they exhibit higher edge strengths. In order to give preference to patches having higher edge strengths, we generate an image I_g consisting of normalized gradient magnitudes from the preprocessed image I_v. The gradient magnitude along the boundary of the cracked regions may vary and therefore the pixels of a cracked region may not have a unique edge strength. In order to assign a unique edge strength to each cracked region, we intend to identify the regions disconnected by weak gradient magnitudes. For this purpose, we create an image I_m by convolving the image I_g with horizontal, vertical, diagonal and anti-diagonal line filters of size 3×3 with filter masks shown in figure 7.7, and recording the maximum response at each

Algorithm 3 Calculation of tED

% For vectors v_1 and v_2 with lengths $|v_1|$ and $|v_2|$, respectively and δ_t as tolerance value,

% Initialization

$D[0,0] := 0$

for $i := 1$ to $|v_1|$ **do** $D[i,0] := i$ **end for**

for $j := 1$ to $|v_2|$ **do** $D[0,j] := j$ **end for**

% Required operation: substition, insertion or deletion

for $i := 1$ to $|v_1|$ **do**

  for $j := 1$ to $|v_2|$ **do**

    $m_1 := D[i-1, j-1] + C(v_1[i], v_2[j], \delta_t)$

    $m_2 := D[i-1, j] + 1$

    $m_3 := D[i, j-1] + 1$

    $D[i,j] = \min(m_1, m_2, m_3)$

  end for

end for

% Result

return tED $:= D[|v_1|, |v_2|]$

% Comparison function: $C(v_1[i], v_2[j], \delta_t)$

if $|v_1[i] - v_2[j]| \leq \delta_t$ **then**

  $C(v_1[i], v_2[j], \delta_t) := 0$

else

  $C(v_1[i], v_2[j], \delta_t) := 1$

end if

pixel.

In all our experiments we observed that pixels around the boundary of cracked regions have a low non-zero response to the line filter. Because of this, the dis-

0	0	0
1	1	1
0	0	0

(a)

1	0	0
0	1	0
0	0	1

(b)

0	1	0
0	1	0
0	1	0

(c)

0	0	1
0	1	0
1	0	0

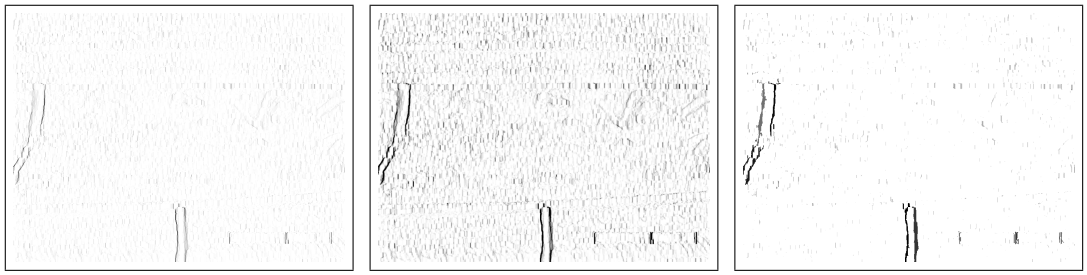
(d)

Figure 7.7: Line filters. (a) Horizontal, (b) main diagonal, (c) vertical and (d) anti-diagonal.

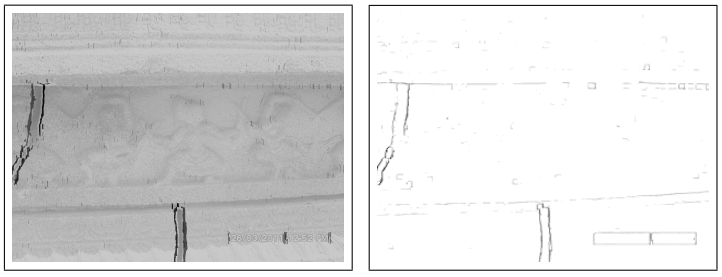
joint cracked regions get connected while performing unique edge strength assignment. To avoid such a situation, the filter responses having lower values are required to be discarded using an image dependent threshold. Since the response to line filters is not expected to vary significantly for pixels in the cracked regions, a threshold with respect to the maximum response can be used. Setting the threshold to 0.1 times the maximum response was found appropriate for discarding the low non-zero responses which were responsible for connecting the disjoint cracked regions. The image I_m is thus updated by discarding the low responses with respect to the maximum response (i.e. setting $I_m(x,y) = 0$ if $I_m(x,y) < 0.1 * max(I_m)$) followed by morphological closing to detect the connected components.

The gradient magnitude image I_g is now updated by using the updated image I_m, such that, the highest gradient magnitude within each connected component is assigned to all the pixels within the respective component. Updating I_g in this manner enables us to assign a unique edge strength value to distinct components. The edge strength image I_e is now constructed by taking the normalized sum of I_g and I_w. For every patch Φ_p for which tED is calculated, we now consider the average of edge strengths of all the pixels within the patches Φ_p, Φ_r and Φ_s, and multiply it with the corresponding tED, to get the weighted tED image I_{tw}. This process is illustrated with an example in figure 7.8.

In order to fill the gap between the boundaries, a morphological closing operation is applied on I_{tw}, with the size of the structuring element depending on the size of the preprocessed image I_v. The morphologically closed image I_{tw} is now multiplied with the resized version of the weight matrix I_w to obtain an intermediate image I_{wc}. In order to assign unique values to different objects for



(a) Initial I_g (b) Refined and closed I_m (c) Updated I_g



(d) I_e (e) I_{tw}

Figure 7.8: Edge strength I_e and weighted tolerant edit distance images I_{tw}. Sizes of I_g, I_m and I_e are the same as that of I_0, while I_{tw} and I_{tED} are of the same size. Here, enlarged and intensity inverted version of I_{tw} is shown for clarity.

segmentation in image I_{wc}, we employ the method used earlier for updating the gradient magnitude image I_g described in the previous paragraph. Thus, by convolving the intermediate image I_{wc} with the 3×3 line filters, thresholding the maximum response image and applying the morphological closing operation, we obtain the image I_c of size $(\frac{M}{m} - 1) \times (\frac{N}{n} - 1)$, as shown in figure 7.9(a) in which the connected components have unique values.

7.1.4 Thresholding

Higher the value of a region in the image I_c, more likely it is to be a crack. Thus, the regions with values lower than a threshold T need to be discarded. Let V denote the array consisting of k unique values in I_c arranged in ascending order. Then, inspired by the threshold selection method for matching features of the scale invariant feature transform (SIFT) given in [88], we estimate the threshold T using algorithm 4.

The image I_c is now updated by setting values less than T to zero. Each pixel in I_c corresponds to an $m \times n$ overlapping patch in I_v. We obtain an initial crack

Algorithm 4 Selection of threshold T

% For an array V consisting of k unique values in I_c arranged in ascending order,

% Initialize
$T := V[k]$

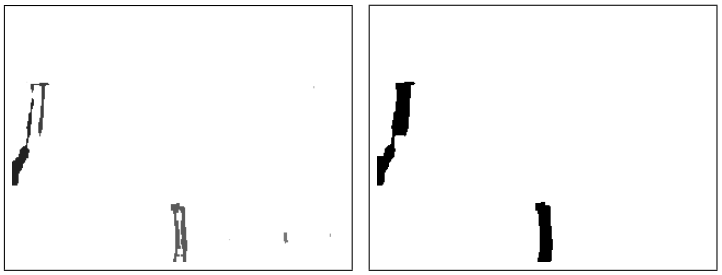
% Update
for $i := k - 1$ to 1 **do**
  if $V[i] < 0.2$ **then**
    break
  end if
  if $(\frac{V[i]}{V[i+1]}) \geq (\frac{V[i-1]}{V[i]})$ **then**
    $T := V[i]$
  end if
end for

% Result
return T

detected image I_1 which is of the same size as that of I_v by copying pixels values from I_c to corresponding patches in I_1. A second morphological closing operation is now applied on the binary image I_1 in order to avoid splitting of the detected region. Note that the image I_1 as shown in figure 7.9(b) gives a good estimate of the cracked regions. However, few pixels of the cracked regions which are similar to the surroundings may still remain undetected. Therefore, a refinement step is required to achieve a more accurate detection.

7.1.5 Refinement

The method described above relies on patch-based comparison which localizes the cracked regions in the binary image I_1. In order to perform a more accurate



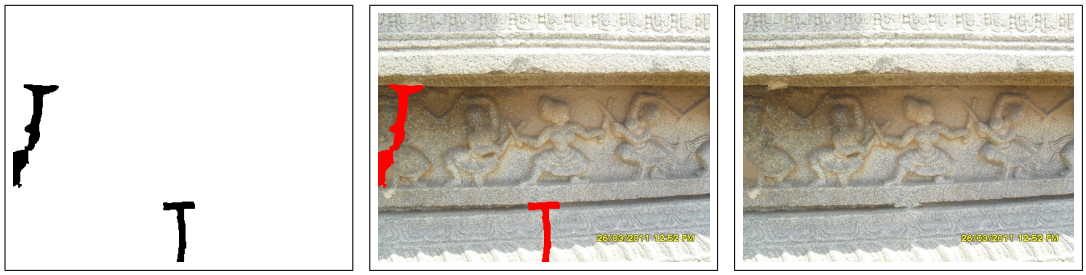
(a) I_c before update (b) I_1

Figure 7.9: Initial detection. Image I_c is thresholded and mapped to I_v to obtain I_1. Size of I_c is same as that of I_{tED}, while I_1 and I_v are of the same size. Here, enlarged, intensity inverted version of I_c is shown for clarity.

detection at pixel level, sophisticated techniques are required, such that, a binary segmentation-based refinement around the initially detected regions can be performed. Interactive image segmentation techniques based on curve evolution [19] and graph-cut optimization [120] have been widely used for accurately detecting roughly marked objects. These segmentation techniques require the user to manually select a region around the object of interest. By optimizing an energy function, the selection is refined to fit the object boundary. The initially detected binary image I_1, which is detected automatically without any user interaction, can be used as an input to the above mentioned interactive segmentation techniques.

For refining I_1, we use the method based on active contours[1] proposed in [19], to obtain the final crack detected binary image I_f, an example of which is shown

[1]For active contour segmentation technique, we have used the implementation available at http://www.mathworks.in/matlabcentral/fileexchange/23847-sparse-field-methods-for-active-contours



(a) I_f (b) I_f overlapped on input image (c) Inpainted image

Figure 7.10: Refinement of the initially detected cracks. (a) Final detection binary image I_f, (b) detected regions overlapped on the input image, (c) inpainted result.

Algorithm 5 Summary of steps involved in proposed crack detection approach.

1: Obtain the weight image I_w from the intensity normalized input image I_0 using Eq. (7.1).
2: Use I_0 and I_w to obtain the preprocessed image I_v.
3: Compute the tolerant Edit Distance image I_{tED} by comparing all the non-overlapping adjacent patches in I_v for similarity as shown in figure 7.6.
4: Generate image I_g consisting of the normalized gradient magnitudes from I_v.
5: Convolve I_g with horizontal, vertical, diagonal and anti-diagonal line filters of size 3×3 and record the maximum response at each pixel to create an image I_m.
6: Discard the low responses in I_m and perform morphological closing to detect the connected components.
7: Update I_g using I_m such that the highest gradient magnitude within each connected component is assigned to all the pixels within the respective component.
8: Take the normalized sum of I_g and I_w to construct image I_e.
9: Multiply I_e with I_{tED} to get the weighted tED image I_{tw} and apply morphological closing to fill gaps.
10: Multiply I_{tw} with the resized version of the weight matrix I_w to obtain an intermediate image I_{wc}.
11: Perform steps 5–7 considering I_{wc} in place of I_g to obtain the image I_c having unique values for the connected components.
12: Update I_c by setting values less than an automatically obtained threshold T to zero.
13: Obtain the initial crack detected binary image I_1 using I_c.
14: Refine I_1 using the method in [19] to obtain the final crack detected image I_f. The regions detected in I_f are inpainted using the technique in [26].

in figure 7.10(a). Thus, we use automatically detected I_1 in place of the interactive input for segmentation.

7.1.6 Inpainting

In order to justify the suitability of the proposed method for inpainting, we also show the inpainted result in figure 7.10(c). Here, we consider the detected cracked regions in the image I_f as the missing regions to be inpainted, which are marked with red color that overlaps the input image as shown in figure 7.10(b). Inpainting of the missing regions is then demonstrated using the technique proposed in [26].

A summary of the steps involved in this approach are given in algorithm 5. We now proceed towards discussing the experimental results in the following subsection.

7.1.7 Experimental results

In our experiments, we show the results for five input images of size 684×912 captured from the world heritage site at Hampi, Karnataka in India and one input image downloaded from the Internet [53] to illustrate the detection and inpainting of multiple cracked regions. We considered patches Φ_p, Φ_r and Φ_s of size 3×3 in our experiments. Use of patches having larger sizes did not significantly improve the detection. In calculation of the tolerant edit distance, we have set the tolerance value $\delta_t = 10$ based on the following experimentation.

We considered many patches at the boundary of known cracked regions from

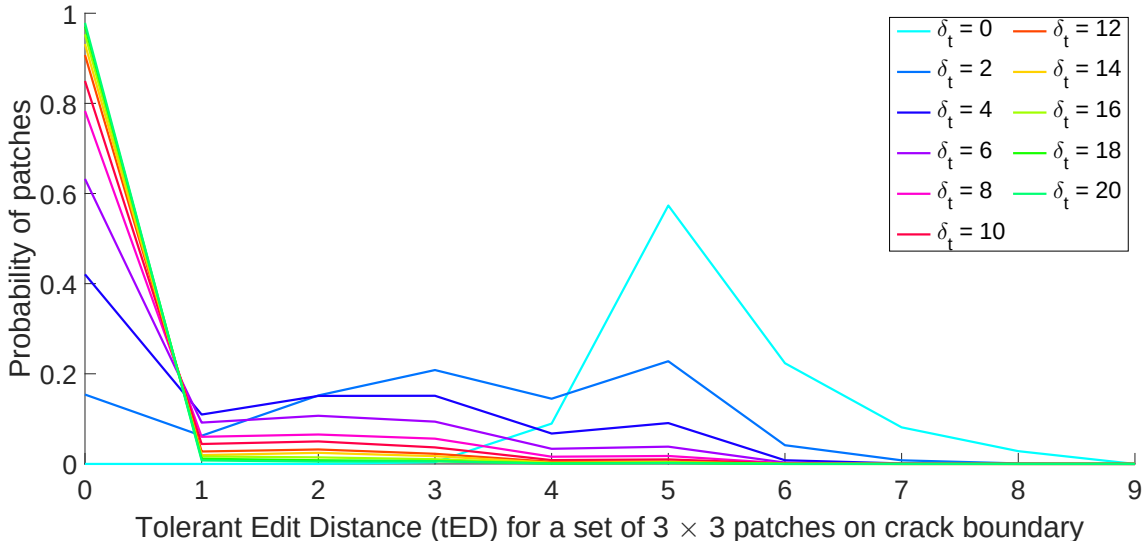
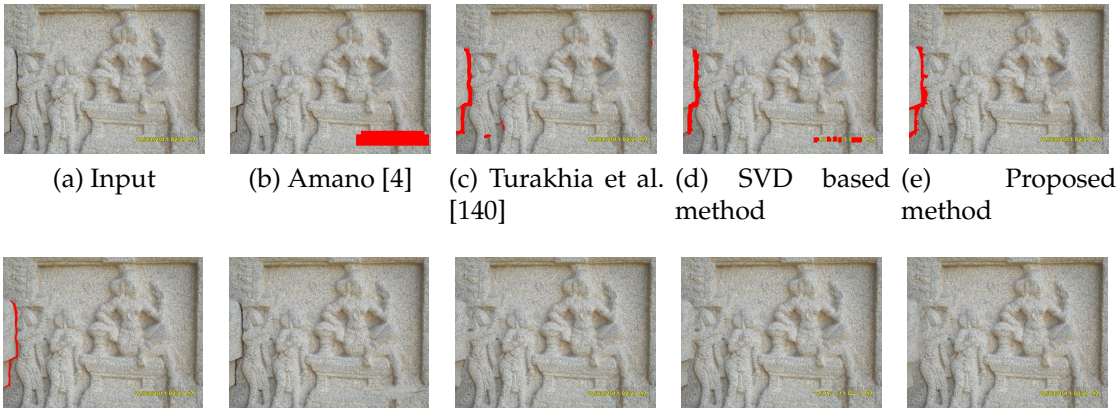


Figure 7.11: Curves for varying tolerance values δ_t.

a number of images, along with their corresponding non-overlapping adjacent patches. For each of these patches, we calculated the tolerant edit distances by varying values of δ_t. Curves of tolerant edit distance versus probability of patches, corresponding to every δ_t were plotted, as shown in figure 7.11. Since the patches belonged to crack boundaries, we have higher edit distance (i.e. $\delta_t = 0$). Increasing the value of δ_t reduces the sensitivity and therefore only large variations can be detected. It is observed that for $\delta_t = 10$, sufficiently large variations were detected and further increasing δ_t did not change the curve significantly. The size of structuring element for morphological closing used for filling in large gaps, depends on the image size. For an image of size $M \times N$, the size of structuring element is chosen as $(\max(M,N)/360 + \min(M,N)/270)$ such that $M, N > 270$.

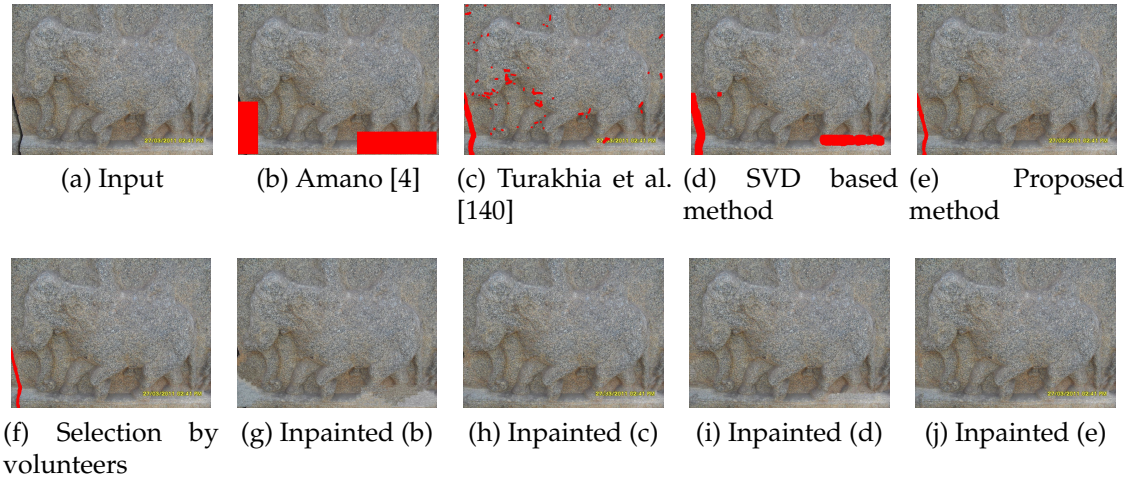
We now show in figures 7.12–7.17, a comparison of the results obtained using the proposed approach, with those obtained using the techniques in [4, 140] and the SVD based crack detection method discussed in section 6.2. It may be noted that the results for the technique in [4] are obtained after fine-tuning the parameters. We show the input images containing cracked regions in figures 7.12(a)–7.17(a). Since the true cracked regions were unavailable, used regions marked by volunteers as the ground truth for comparison. These are shown by the re-



(a) Input (b) Amano [4] (c) Turakhia et al. [140] (d) SVD based method (e) Proposed method

(f) Selection by volunteers (g) Inpainted (b) (h) Inpainted (c) (i) Inpainted (d) (j) Inpainted (e)

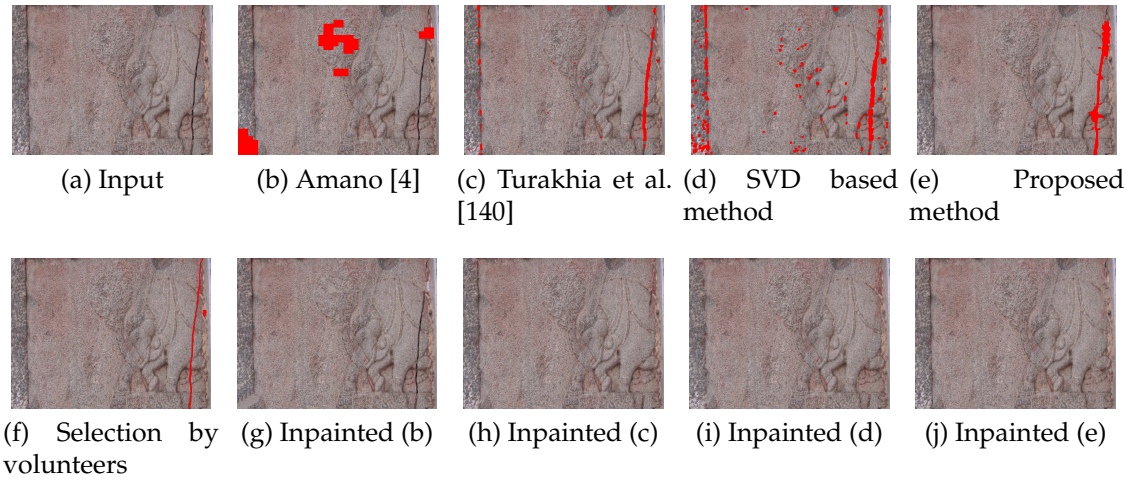
Figure 7.12: Detection and inpainting of a cracked region in an image of a wall carving containing people. The detected cracked pixels in (b)–(e) and those marked by volunteers in (f) are shown in red color. Inpainted version the detected crack in (b)–(e) is shown in (g)–(j), respectively.



(a) Input (b) Amano [4] (c) Turakhia et al. [140] (d) SVD based method (e) Proposed method

(f) Selection by volunteers (g) Inpainted (b) (h) Inpainted (c) (i) Inpainted (d) (j) Inpainted (e)

Figure 7.13: Detection and inpainting of a narrow cracked region near the bottom left corner of the image. The detected cracked pixels in (b)–(e) and those marked by volunteers in (f) are shown in red color. Inpainted version of the detected crack in (b)–(e) is shown in (g)–(j), respectively.

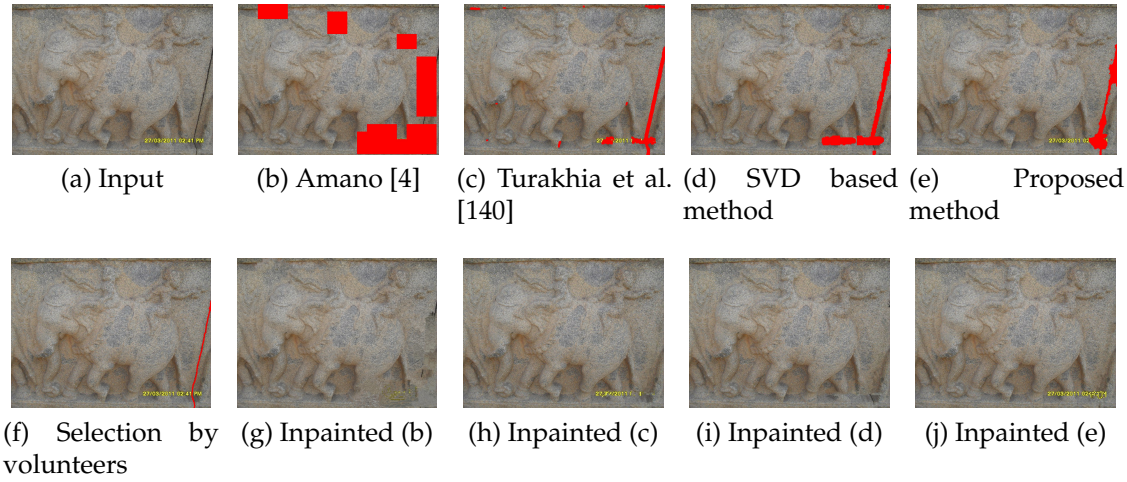


(a) Input (b) Amano [4] (c) Turakhia et al. [140] (d) SVD based method (e) Proposed method

(f) Selection by volunteers (g) Inpainted (b) (h) Inpainted (c) (i) Inpainted (d) (j) Inpainted (e)

Figure 7.14: Detection and inpainting of cracked regions across an artistic work. The detected cracked pixels in (b)–(e) and those marked by volunteers in (f) are shown in red color. Inpainted version the detected cracks in (b)–(e) are shown in (g)–(j), respectively.

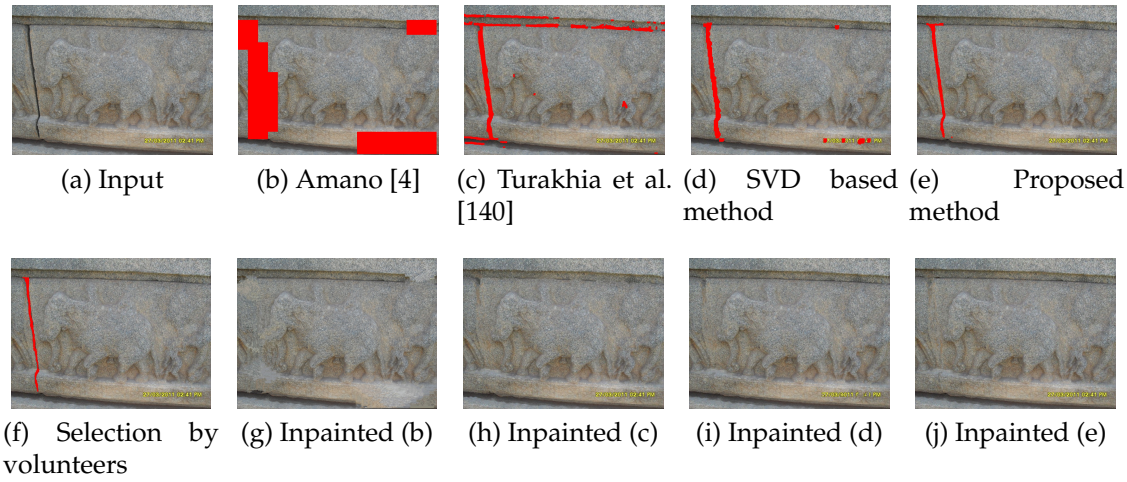
gions marked in red color figures 7.12(f)–7.17(f). The results for crack detection using the technique proposed in [4] are shown in figures 7.12(b)–7.17(b), while those using the technique proposed in [140] are shown in figures 7.12(c)–7.17(c). We also compare the results obtained using the SVD based crack detection approach discussed in section 6.2 that we show in figure 7.12(d)–7.17(d) with those obtained using our proposed method illustrated in figures 7.12(e)–7.17(e). Here,



(a) Input (b) Amano [4] (c) Turakhia et al. [140] (d) SVD based method (e) Proposed method

(f) Selection by volunteers (g) Inpainted (b) (h) Inpainted (c) (i) Inpainted (d) (j) Inpainted (e)

Figure 7.15: Detection and inpainting of a cracked region in an image containing multiple textured regions. The detected cracked pixels in (b)–(e) and those marked by volunteers in (f) are shown in red color. Inpainted version of the detected crack in (b)–(e) is shown in (g)–(j), respectively.



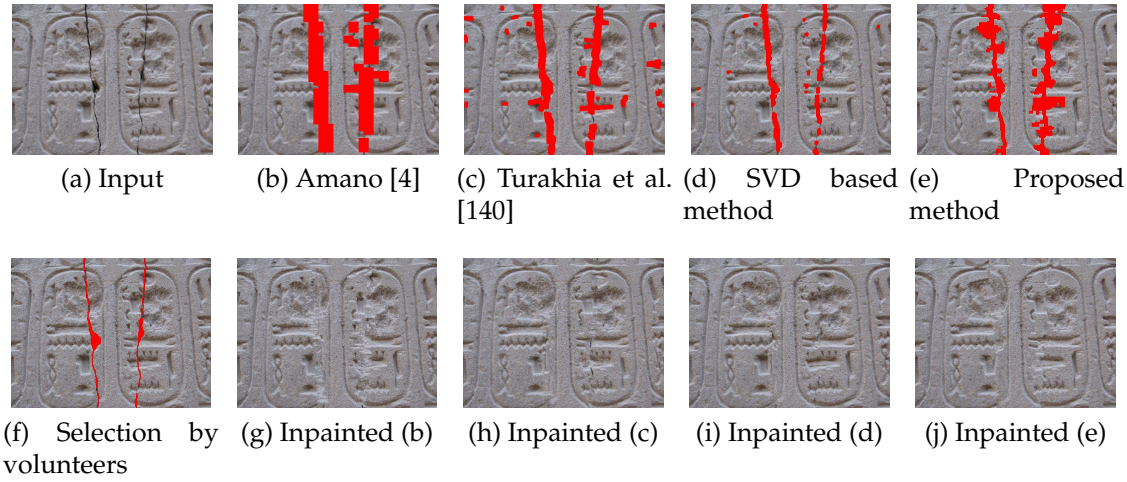
(a) Input (b) Amano [4] (c) Turakhia et al. [140] (d) SVD based method (e) Proposed method

(f) Selection by volunteers (g) Inpainted (b) (h) Inpainted (c) (i) Inpainted (d) (j) Inpainted (e)

Figure 7.16: Detection and inpainting of an elongated cracked region. The detected cracked pixels in (b)–(e) and those marked by volunteers in (f) are shown in red color. Inpainted version of the detected crack in (b)–(e) is shown in (g)–(j), respectively.

in order to demonstrate the suitability for automating the repair of cracked regions, we also show the corresponding inpainted images obtained by using the method proposed in [26].

Figure7.12 shows the image of a wall carving containing people. Here, the techniques [4, 140] and SVD based method detect more regions than the ground truth. On the other hand, the region detected using the proposed method is simi-



(a) Input (b) Amano [4] (c) Turakhia et al. [140] (d) SVD based method (e) Proposed method

(f) Selection by volunteers (g) Inpainted (b) (h) Inpainted (c) (i) Inpainted (d) (j) Inpainted (e)

Figure 7.17: Detection and inpainting of multiple cracked regions. The detected cracks in (b)–(e) and those marked by volunteers in (f) are shown in red color. Inpainted version of the detected cracks in (b)–(e) are shown in (g)–(j), respectively.

lar to that marked by the volunteers. Similar is the case for results considering an image containing a narrow cracked region shown in figure 7.13.

In figure 7.14 we show a sensitive case where the cracked region appears across an artistic work. Here, the method in [4] performs poor detection with the result using the method in [140] fails to detect significant number of cracked pixels due to which the cracked regions are visible after inpainting. In this case, the SVD based method performs better detection of cracked region, yet, it also detects many smaller regions that are not marked by the volunteers. Here, again, our proposed method perform even better and detects the region similar to the ground truth which can also be noticed from the plausibly inpainted result. Similarly, for an image containing multiple textured regions shown in figure 7.15, the proposed method performs visually better detection of cracked region in comparison to the methods in [4, 140] and SVD based crack detection method.

In figure 7.16 we show the detection of an elongated cracked region. Here, one can observe that the method in [4] detects large blocks around the cracked region marked by the volunteers. Also, the method in [140] and the SVD based method detect more number of regions in comparison to the ground truth. On the other hand, our proposed method perform better detection of the cracked regions even in this case. Likewise, for an image containing multiple cracked regions

shown in figure 7.17, our proposed method preforms more accurate detection of the cracked regions, in comparison to the methods [4, 140] and SVD based crack detection method.

Input image	# Cracked pixels	Defect detection method [4]			Method by Turakhia et al. [140]			SVD based method			Proposed method		
		Recall	Precision	Time (sec)	Recall	Precision	Time (sec)	Recall	Precision	Time (sec)	Recall	Precision	Time (sec)
Figure 7.12(a)	3494	0.000	0.000	109.0	0.988	0.887	21.65	0.953	0.743	04.51	0.990	1.000	03.62
Figure 7.13(a)	3819	0.918	0.370	13.02	0.970	0.390	22.22	1.000	0.422	14.54	0.969	1.000	03.32
Figure 7.14(a)	5162	0.046	0.068	302.3	0.749	0.678	23.29	0.863	0.392	12.77	0.840	0.997	05.02
Figure 7.15(a)	2997	0.783	0.737	12.06	0.999	0.728	25.16	0.921	0.678	04.80	0.990	0.997	03.49
Figure 7.16(a)	5435	1.000	0.579	19.01	0.974	0.974	29.92	0.987	0.857	04.89	0.985	0.996	04.77
Figure 7.17(a)	2276	0.966	0.949	1500	0.932	0.949	13.64	0.808	0.898	05.23	0.952	0.989	07.44

Table 7.1: Comparison in terms of recall and precision for images shown in figures 7.12–7.17.

We now perform an objective comparison of the results discussed above. For this purpose the popularly known recall and precision metrics defined in equation (6.7) are considered. However, for showing an insight of the robustness of our proposed algorithm, we use a slightly different precision measure defined as,

$$Precision = \frac{|Ref_{conn}|}{|Dect|}, \tag{7.2}$$

where, $Dect$ are the pixels detected by the algorithm to be in the cracked regions and Ref_{conn} are those pixels detected in $Dect$ that are connected to cracked regions marked by volunteers.

The quantitative measures recall and precision for results displayed in figures 7.12–7.17 are given in table 7.1. From the table we observe that both recall and precision values for the detected cracked regions using our method are nearer to 1, indicating that the desired cracked pixels have been detected with high accuracy. Thus, our method performs better crack detection. On the other hand, the

technique proposed in [4] results in detection of either (a) pixels that do not correspond to the desired cracked regions or (b) too many pixels around the desired cracked regions. The later leads to unnecessary inpainting of many regions that modifies large undamaged regions in the image, which is not desirable. Moreover, it slows down the inpainting as the inpainting process is computationally expensive. The results of our crack detection method are at par with and in some cases better than those obtained using the technique in [140]. Yet, our proposed method is significantly faster, more accurate and the inpainted results are convincing. Also, in comparison to our previous crack detection approaches discussed in chapter 6, the recall values for the proposed approach are similar but the precision values show significant improvement indicating higher accuracy of detection.

The implementation details along with the timing information are presented as follows. For images, the calculation of tolerant edit distance which involves comparison of many patches is implemented in C (Matlab MEX) while the rest of the method is implemented in Matlab. For a 684×912 sized image, the initial detection takes about 1.5 seconds on a Windows 7 Professional operating system with 32 bit Intel Core i5, 2.5GHz CPU and 3 GB RAM. The remaining detection time is spent on refinement, which again is a C (Matlab MEX) implementation. However, in the same setup, the process for inpainting (for example the regions detected in figure 7.12(e)) requires about 37 seconds, which is also a C (Matlab MEX) implementation. Therefore, at present the implementation does not execute in real-time and needs to be performed offline. In future, if a faster inpainting method is developed, the implementation could run in nearly real-time.

Here, we have discussed a crack detection technique based on tolerant edit distance for performing auto-inpainting. This technique shows substantial improvement in the accuracy of detection in comparison to other approaches. In following section, we extend this approach to perform auto-inpainting in videos and also provide a measure to quantify the quality of inpainted videos.

7.2 Extension to Auto-inpaint Videos with Quantitative Assessment

In order to extend the approach of crack detection and inpainting discussed in section 7.1 to videos, one may think of performing frame-by-frame detection and inpainting. This abstraction, however, in practice is a long-drawn-out process as it does not exploit the inter-frame redundancy. Also, there may be occlusion or change in illumination across frames as the camera moves, due to which the pixels corresponding to cracked regions detected in one frame may not map to the pixels corresponding to the same cracked regions detected in some other frame. Since the inpainting task is highly sensitive to the pixels to be inpainted, it leads to large variations in the two inpainted frames for the same cracked regions. As a result, the auto-inpainted videos created by detecting and inpainting frame by frame, appear unstable and the effect of seam becomes visible.

Alternatively, one may use motion as a cue to track and inpaint the cracked regions across subsequent frames. Motion estimation and compensation have been popularly used in video compression techniques [67, 128]. Here intermediate frames are generated using independent frames and motion parameters. However, since these methods are block based, their use to inpaint videos results in blocking artefacts. Moreover, they are computationally expensive and the motion parameters are estimated using 2D-2D transformation. A frame-to-frame transformations is, therefore, needed to track the damaged regions in subsequent frames for creating a seamlessly inpainted video.

Brown and Lowe [13] have suggested a method for automatic image stitching, wherein transformation between the images to be stitched is calculated by matching keypoints invariant to rotation, scaling and view point. Here, the transformation is considered to be projective or a homography [58]. Since the videos captured at heritage sites usually contain nearly planar rigid objects / scene with a moving camera, we can consider the video frames to be images captured from different viewpoints. Hence, the transformation between these frames can be represented by a homography.

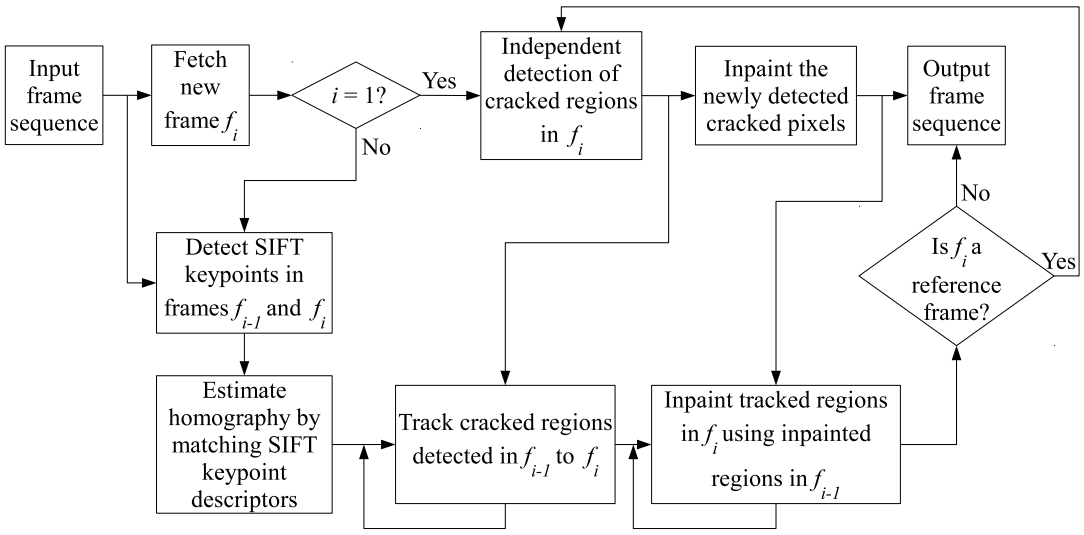


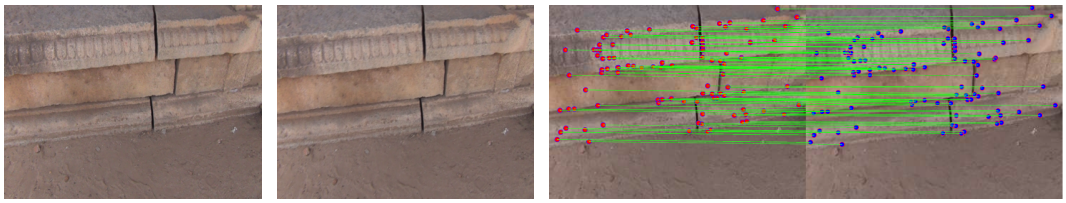
Figure 7.18: Our approach for extending crack detection and inpainting in images to videos of heritage scenes.

Our approach for detecting and inpainting the cracked regions in videos is shown in figure 7.18. Here, we consider pairs of temporally adjacent frames and use the homography [58] to track the cracked regions from one frame to another. The first video frame is initially considered as the reference frame, which is later updated based the camera movement. The cracked regions are detected in reference frames using the method described in section 7.1 and then tracked to subsequent frames. Similarly, the detected cracks are inpainted in the reference frames using the technique proposed in [26] and then mapped to the tracked regions in the subsequent frames. Note that the inpainting of video frames cannot be done by simply copying objects visible in other frames, as done in [111]. This is because, an object to be inpainted in one frame also needs to be inpainted in other frames as well, which mandates the use of a hole filling technique. Details of the proposed approach are given below.

7.2.1 Homography estimation

As already mentioned, since the videos captured at heritage sites usually contain nearly planar rigid objects / scene with a moving camera, we consider the video frames to be images captured from different viewpoints. Hence, the transformation between these frames can be represented by a homography [58, 77], which

we estimate by extracting keypoints and matching their scale invariant feature transform (SIFT) descriptors[2] [88].



(a) (b) (c)

Figure 7.19: Matching of SIFT keypoints. (a)–(b) Two frames of a video; (c) Pairs of matching keypoints shown by green joining lines.

Let the keypoint at location (x_1, y_1) in the first frame match the keypoint at location (x_2, y_2) in the second frame. For a set of such matching keypoints, the homography matrix H obeys the following relationship [58],

$$\begin{bmatrix} x_2' \\ y_2' \\ z_2' \end{bmatrix} = H \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}, \tag{7.3}$$

where (x_2', y_2', z_2') are the homogeneous coordinates for the point (x_2, y_2) in the second frame such that $x_2 = \frac{x_2'}{z_2'}$ and $y_2 = \frac{y_2'}{z_2'}$, and H is a 3×3 non-singular matrix. Using the set of matched keypoint locations, the homography matrix H is estimated using equation (7.3) by setting $z_2' = 1$ i.e. setting the homogeneous coordinates $(x_2', y_2', z_2') = (x_2, y_2, 1)$. This estimation of the matrix H is done by a random sampling consensus (RANSAC) [43] of all the matching keypoints at locations (x_1, y_1) and (x_2, y_2) in the two frames that obey the relationship[3] $[x_2, y_2, 1]^T = H[x_1, y_1, 1]^T$. Figure 7.19 illustrates the matching of SIFT keypoints between a pair of video frames.

[2] An implementation for extraction and matching of SIFT keypoints and corresponding descriptor is available at http://www.cs.ubc.ca/~lowe/keypoints/

[3] For fitting homography to keypoints using RANSAC, we used the code available at http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/Robust/ransacfithomography.m

7.2.2 Reference frame detection

A reference frame is the one in which cracked regions are detected independently. While capturing the video with a moving camera, new cracked regions may appear. If the cracked regions are detected in the first frame and tracked across all subsequent frames, the new cracks that appear as the camera moves will not be detected. Therefore, an independent crack detection needs to be performed quasi-periodically depending the camera movement. Thus, for fast camera movement, the detection needs to be performed more frequently, while for slow camera motion, a less frequent detection is required. If the camera motion can be measured, an appropriate threshold can be set to declare an incoming frame as a reference frame. An intuitive way to quantify the camera motion is to calculate the magnitude of translation.

The authors in [41, 91] have shown that, given a homography matrix, it can be decomposed to estimate the translation. The decomposition yields four solutions in general out of which only two are physically possible. However, each of these solutions has the same magnitude of translation. We make use of this information to detect the reference frame. The solutions for decomposition[4] of a homography H are obtained using the method in [91].

Let t be the translation vector of one of the obtained solutions, such that, $t = [t_1, t_2, t_3]^T$. Then the magnitude of translation is given by $|t| = \sqrt{t_1^2 + t_2^2 + t_3^2}$. Also, let δ_r be the threshold for translation. Considering the first video frame as a reference *ref*, a homography along with the translation between the reference and every incoming frame f_i is calculated. If the magnitude of translation is above a threshold δ_r, then the incoming frame f_i is declared to be a reference frame. For the new incoming frames, f_i becomes the reference frame. This method for detecting the reference frames in a videos is given in algorithm 6.

For selection of the translation threshold δ_r to detect the reference frames, we conducted the following experiment. We manually selected two frames viz. (1) the frame in which a cracked region has completely appeared and (2) the frame

[4]For decomposition of estimated homography, we have used the implementation available at http://cs.gmu.edu/~kosecka/examples-code/homography2Motion.m

Algorithm 6 Detection of reference frame

% Let the i^{th} video frame be denoted by f_i, such that the video consists of total k frames. If $R_i := 1$ then f_i is a reference frame.

% Initialization

$R_1 = 1; R_i := 0 \; \forall i := 2, \ldots, k.$

$ref := f_1$. {reference frame.}

% Update R_i

for $i := 2$ to k **do**

  $suc := f_i$. {subsequent frame.}

  Estimate translation t between ref & suc.

  if $|t| \geq \delta_r$ **then**

    $R_i := 1$.

    $ref := suc$.

  end if

end for

% Result

return $R_i \; \forall i := 1, \ldots, k.$

in which the next cracked region begins. For every such pair of frames, translation was calculated. Conducting the experiment on a number of videos having frames of size 270×360 revealed that the average value of $\delta_r = 25$ can be used to detect new incoming cracked regions. However, the problem with this threshold is that, while a part of the newly appearing cracked region gets detected successfully, the remaining part which appears in subsequent frames is not detected. For successful detection of the complete cracked regions, a lower value of threshold is required. By keeping the threshold from 25 to 0, we found $\delta_r = 5$ to be an appropriate threshold for successful detection of the complete cracked regions. Also note that the intensity change in corresponding pixels across the frames within this small translation is negligible. This enables a seamless copying of pixel val-

ues when propagating the already inpainted cracked regions across subsequent frames. We have, therefore, set $\delta_r = 5$.

Having discussed homography estimation and reference frame detection, we now proceed with the discussion of tracking and inpainting the cracked regions across frames in the following section.

7.2.3 Tracking and inpainting cracked regions across frames

For a pair of temporally adjacent frames f_{i-1} and f_i, the locations (x_i, y_i) of cracked pixels in f_i can be tracked from the frame f_{i-1} using the corresponding locations (x_{i-1}, y_{i-1}) as follows,

$$\begin{bmatrix} x_i^{'} \\ y_i^{'} \\ z_i^{'} \end{bmatrix} = H_i \begin{bmatrix} x_{i-1} \\ y_{i-1} \\ 1 \end{bmatrix}, \tag{7.4}$$

where $(x_i^{'}, y_i^{'}, z_i^{'})$ are the homogeneous coordinates for the point (x_i, y_i) such that $x_i = \frac{x_i^{'}}{z_i^{'}}, y_i = \frac{y_i^{'}}{z_i^{'}}$ and H_i denotes the homography between frames f_{i-1} and f_i. Here, it may happen that estimated coordinates x_i and y_i are real numbers. These are rounded to the nearest integers so that we have the tracked pixels at integer locations. For simplicity, let the integer-rounded location coordinates be denoted by (x_i, y_i). Setting these cracked pixel locations to 1 with all other locations set to a value of 0, a crack-mask consisting of 1's and 0's is constructed for the frame f_i. Since homography introduces geometric distortions, it may happen that some narrow cracked regions detected in the frame f_{i-1} may become disjoint regions in the newly constructed crack-mask, which leads to some part of the cracked regions being missed out. In order to avoid this, we use morphological closing on the crack-mask to connect the nearby disjoint regions. The crack-mask now gives the locations of the tracked cracks in the frame f_i that correspond to the crack regions detected in the frame f_{i-1}. Figure 7.20 illustrates the tracking of cracked regions.

We now describe how an incoming frame is processed. The first video frame f_1 being a reference frame is independently inpainted after identifying the cracked

(a) (b) (c)

Figure 7.20: Tracking detected regions using the estimated homography matrix. (a) Detected damaged regions in the frame f_{i-1}; (b) frame f_i; (c) tracked cracks in the frame f_i. Green lines show the mapping of few points on the boundary of crack regions, while the detected and tracked cracked regions using SIFT features are shown in red.

regions in it. Any subsequent incoming frame f_i may or may not be a reference frame depending on the camera motion. For both cases, we use the above procedure along with equation (7.4) to track cracked regions from f_{i-1} to f_i. Let P_i denote the binary image consisting of the cracked regions tracked from frame f_{i-1} to frame f_i. In case f_i is not a reference frame, it can be inpainted by filling up the location of the tracked crack pixels (i.e. $\{(x_i, y_i)|P_i(x_i, y_i) = 1\}$). This is achieved by simply copying the values of the corresponding pixels from the inpainted version of the previous frame f_{i-1}. Note that the frames are temporally adjacent and the change in intensity of corresponding pixels is negligible. Also note that the selected translation threshold δ_r is small enough so that the change in intensity of corresponding pixels across frames within this translation is also negligible. Thus, the copying of pixel values across subsequent frames does not introduce any seam.

Since the homography matrix H_i is non-singular, its inverse H_i^{-1} exists. Therefore, the crack pixels at locations (x_i, y_i) and the corresponding locations (x_{i-1}, y_{i-1}) from the previous frame f_{i-1}, must be related as follows,

$$\begin{bmatrix} x'_{i-1} \\ y'_{i-1} \\ z'_{i-1} \end{bmatrix} = H_i^{-1} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}, \tag{7.5}$$

where $(x'_{i-1}, y'_{i-1}, z'_{i-1})$ are the homogeneous coordinates for the point (x_{i-1}, y_{i-1}), such that $x_{i-1} = \frac{x'_{i-1}}{z'_{i-1}}$ and $y_{i-1} = \frac{y'_{i-1}}{z'_{i-1}}$. Since x_i and y_i were rounded to integers,

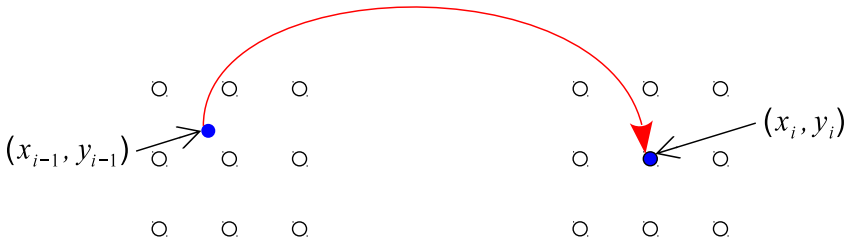


Figure 7.21: Example showing a source pixel at a non-integer location (x_{i-1}, y_{i-1}) in the frame f_{i-1}, tracked to a pixel at location (x_i, y_i) in the frame f_i. The circles with black boundary indicate the integer locations.

we may obtain the corresponding x_{i-1} and y_{i-1} as real numbers as illustrated in figure 7.21. The intensity at this location is obtained by considering the first-order integer location neighborhood and using the bilinear interpolation. It may be noted that inpainting performed in this manner across is almost insensitive to small changes in the morphologically closed crack-mask due to directly copying the values from the previously inpainted regions.

If the incoming frame f_i is a reference frame, then crack detection is performed independently. However, since only the newly appearing cracked pixels need to be inpainted, we first calculate the binary image P_i consisting of the cracked regions tracked from the previous frame f_{i-1}. Now, let B_i denote the crack detected binary image corresponding to f_i obtained using the method discussed in section 7.1. Then, the binary image Q_i consisting only the newly appearing cracked pixels is given by,

$$Q_i(x_i, y_i) = \begin{cases} 1, & B_i(x_i, y_i) - P_i(x_i, y_i) > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{7.6}$$

Now, an initial inpainting of the new reference frame f_i is done by using the inpainted version of the previous frame f_{i-1} and the binary image P_i. The locations (x_{i-1}, y_{i-1}) in frame f_{i-1} corresponding to pixels at locations $\{(x_i, y_i) | P_i(x_i, y_i) = 1\}$ are obtained using the relation in equation (7.5). Similar to inpainting a non-reference frame as described above, the pixels at locations (x_i, y_i) are filled by copying values from the corresponding pixels at locations (x_{i-1}, y_{i-1}) to obtain the initial inpainted image. The newly detected cracked pixels given by the binary image Q_i are the holes to be filled in the initially inpainted image. The final

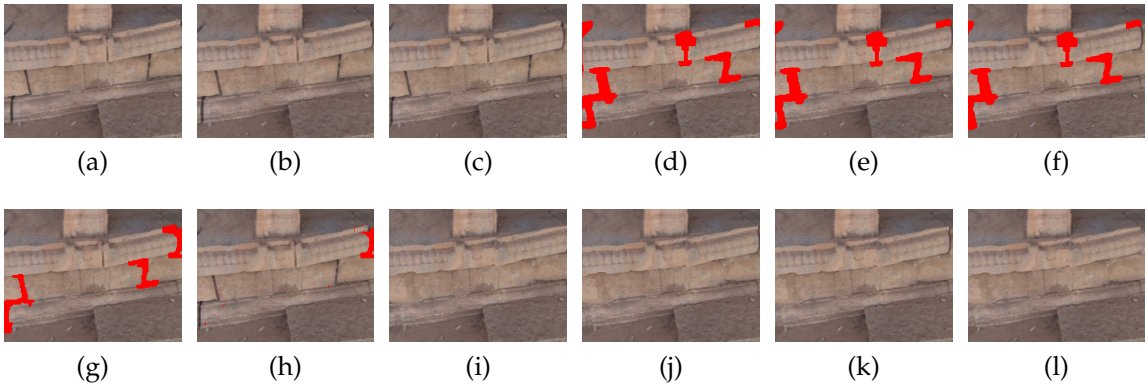


Figure 7.22: Inpainting a newly appearing reference frame f_i. (a),(b),(c) show frames f_{i-2}, f_{i-1} and f_i, respectively; the cracked regions corresponding to (a),(b),(c) tracked from detected cracks in previous frames are shown in (d),(e),(f); independent crack detection in f_i is shown in (g), while the newly appearing cracked pixels in (g) with respect to (f) are displayed in (h); the inpainted versions of f_{i-2}, f_{i-1}, f_i obtained by copying pixels from respective previous inpainted frames are shown in (i),(j),(k); final inpainted version of f_i obtained after inpainting the newly detected pixels is shown in (l). Note that the crack visible near the right side in (k) is filled in (l) by independently inpainting pixels shown in (h).

inpainted version of the reference frame is obtained by using the method proposed in [26] considering the initial inpainted image and the binary image Q_i as inputs. An example for performing inpainting when a new reference frame appears is illustrated in figure 7.22.

It may happen that a detected reference frame is highly blurred or noisy due to an unstable camera motion. In such a case, the crack detection method described in section 7.1 may fail and detect many regions as cracks. This can be avoided by simply thresholding the number of pixels in the newly detected cracked regions. Assuming that the number of pixels in the cracked regions do not vary substantially across the reference frames or whenever a new reference frame is encountered, we set a threshold δ_0 based on the number of cracked pixels detected in the first frame. This is a valid assumption because, while the camera moves and new cracked regions enter a frame, some pixels of the previously detected cracked regions may exit. Also, even if the cracked pixels do not exit, we expect only few new cracked pixels to enter. Therefore, the threshold δ_0 accounting for the newly entering cracked pixels is set to half the number of cracked pixels detected in the first frame.

Algorithm 7 Video frame inpainting

% Let the i^{th} video frame be denoted by f_i, such that the video consists of total k frames. $R_i := 1$ denotes f_i is a reference frame. Let A_i denote the inpainted version of frame f_i.

% Initialization

Detect damaged regions in f_1 to get B_1.

Set threshold $\delta_0 := |B_1|$.

Perform inpainting on f_1 using B_1 to get A_1.

% Loop

for $i := 2$ to k **do**

  Extract SIFT descriptors and homography H_i.

  Calculate P_i by tracking damaged regions.

  if $R_i := 1$ **then**

    Detect damaged regions in f_i to get B_i.

    Calculate Q_i using P_i and B_i.

    if $|Q_i| :\leq \delta_0$ **then**

      Calculate S_i using P_i and B_i.

      Fill pixels $\{(x_i, y_i)|S_i(x_i, y_i) = 1\}$ using A_{i-1} to get initial inpainted image *init*.

      Perform inpainting on *init* using Q_i to get A_i.

    else

      $R_i := 0$, $R_{i+1} := 1$.

      Fill tracked pixels $\{(x_i, y_i)|P_i(x_i, y_i) = 1\}$ using A_{i-1} to get A_i.

    end if

  else

    Fill tracked pixels $\{(x_i, y_i)|P_i(x_i, y_i) = 1\}$ using A_{i-1} to get A_i.

  end if

end for

% Result

return Inpainted frames $A_i \; \forall i := 1, \ldots, k$.

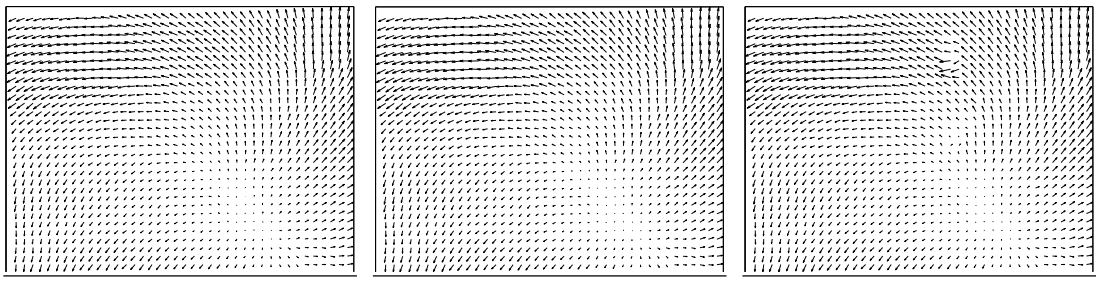
Let $|Q_i|$ denote the number of newly detected cracked pixels in the frame f_i and $|B_1|$ denote the number of cracked pixels detected in the first frame. Then, for a reference frame f_i, if we have $|Q_i| > \delta_0$ (such that $\delta_0 = 0.5 * |B_1|$), the frame f_i is treated as a non-reference frame and inpainting is performed accordingly. Also, the frame f_{i+1} is set as a reference frame, provided f_i is not the last frame. The complete procedure for auto-inpainting video frames is given in algorithm 7.

7.2.4 Measuring temporal consistency of the inpainted video

The quality of a processed video is usually quantified in terms of some metric by comparing the video with an undistorted source. For example, in video compression, the quality of a video reconstructed at a receiver is measured by comparing it with the original video transmitted by the sender. However, in some applications the original source or reference is not available for comparison. Video inpainting is one such application in which missing regions in frames need to be filled up and hence a reference for comparison is not available. In such a case, the objective quantification of the video quality is based on no-reference video quality assessment (NR VQA) measures viz. blockiness, bluriness and sudden local changes [38, 122, 123].

The NR VQA measures listed above estimate the video quality directly from the processed (i.e inpainted) video without considering the input video. Nevertheless, in an application like video inpainting, some information from the unprocessed video also can be used to quantify the quality of the processed video. Intuitively, to obtain a temporally plausible inpainted video, the optical flow of the input video should be maintained on inpainting, provided the objects to be inpainted are stationary. In other words, the optical flow between every pair of temporally adjacent frame in input and corresponding pair of frames in the inpainted video should be similar. The inpainting of only the stationary object is a valid assumption for inpainting videos of heritage monuments. With this cue, the optical flow between every pair of adjacent frames in both input as well as inpainted video can be estimated and used to quantify the quality of the inpainted video. One can estimate the optical flow by using the classic method proposed by

Lucas and Kanade [89].



(a) (b) (c)

Figure 7.23: Optical flow between a pair of temporally adjacent frames in (a) input video, (b) auto-inpainted video using proposed method, (c) video generated by auto-inpainting every frame independently. The optical flow in (a) and (b) appear to be similar while some haphazard orientations in the optical flow are observed in (c).

Consider $L_0(i)$ and $D_0(i)$ be the magnitude and direction, respectively, of the optical flow between the i^{th} and $i+1^{th}$ frames in the input video. Similarly, let $L_1(i)$ and $D_1(i)$ be the magnitude and direction, respectively, of the optical flow between the i^{th} and $i+1^{th}$ frames in the inpainted video. Both L and D are vectorized using lexicographical ordering. Then, the temporal consistency between i^{th} and $i+1^{th}$ frames is given by the Pearson's correlation coefficient $r(i)$ as follows [72].

$$r(i) = \frac{1}{l-1} \sum_{j=1}^{l} \frac{(K_0^j(i) - \bar{K}_0)(K_1^j(i) - \bar{K}_1)}{\sigma_0(i)\sigma_1(i)}, \tag{7.7}$$

where K can be the vector of the magnitude (L) or direction (D) of optical flow, $\bar{K}$ and σ are mean and standard deviation of K respectively, and l represents the length of K. The value $r(i) = +1$ indicates perfect positive correlation, $r(i) = -1$ indicates perfect negative correlation while $r(i) = 0$ corresponds to no correlation between the vectors. The average value of r for all the pairs of adjacent frames then gives the temporal consistency between the input and the inpainted videos. A higher average value of r indicates higher temporal consistency. An example of temporal consistency in terms of optical flow is shown in figure 7.23.

7.2.5 Experimental results

We now present four results of our auto-inpainting method on videos captured by us from the heritage site at Hampi, Karnataka, India. Here, we consider the camera movement to be unconstrained while capturing video of the heritage site, as such videos are typically captured by novices, hobbyists and tourists. The results are shown in figures 7.24–7.27 where we show 6 frames of each video. Although the videos were captured at only one heritage site, our method is generic and should work for other heritage site videos. As an example, we show one more result on a video of the McConkie Ranch Petroglyphs near Vernal, Utah, USA, in figure 7.28 and it demonstrates the effectiveness of the proposed method. This video was uploaded by an enthusiast on the popular streaming site YouTube [124].

From the reported results, we can observe that by using our method, the detected cracked regions are effectively tracked and plausibly inpainted to get a

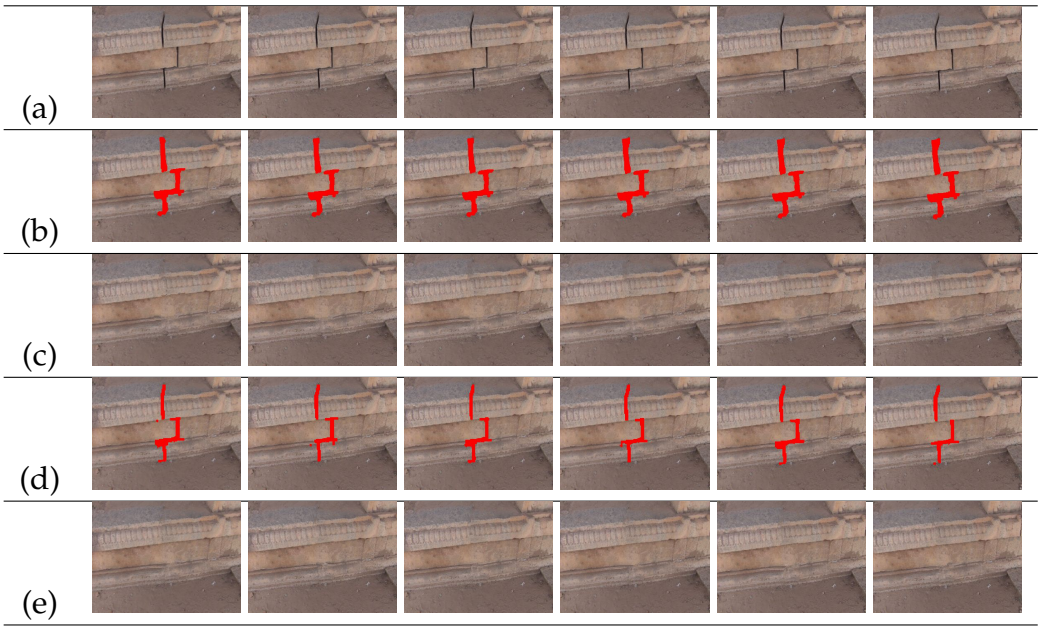


Figure 7.24: Result of auto-inpainting in video frames containing cracked regions in a Hampi wall. (a) Input frame sequence, left most frame is the reference frame; (b) cracked regions detected in the reference frame tracked using proposed method; (c) inpainted frames corresponding to frames in (b); (d) cracked regions detected independently in every frame; (e) inpainted frames corresponding to frames in (d).

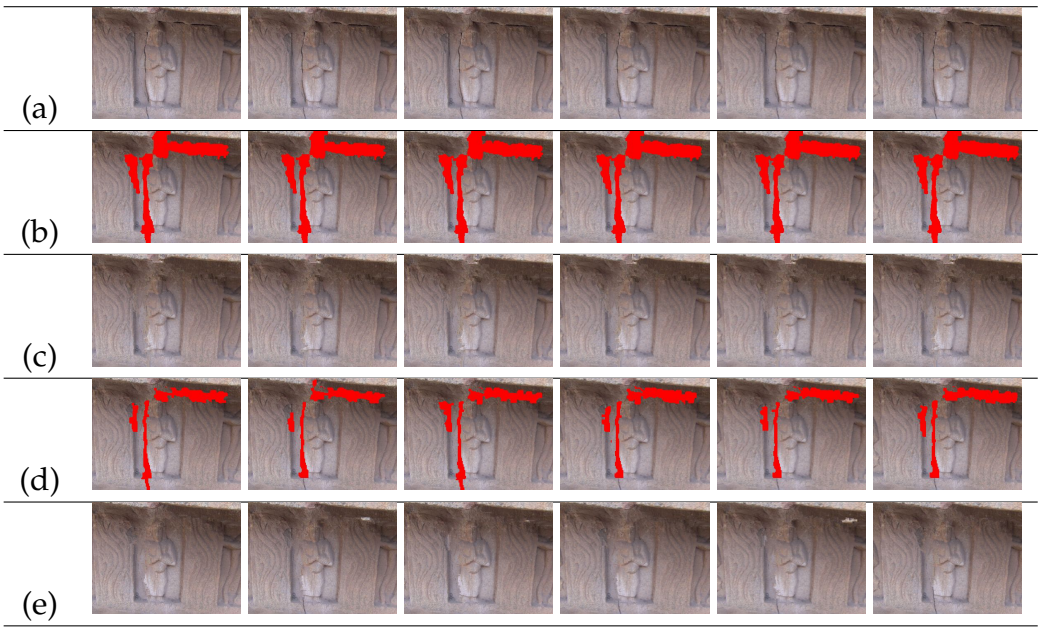


Figure 7.25: Result of auto-inpainting in video frames containing cracked regions around stone carving. (a) Input frame sequence, left most frame is the reference frame; (b) cracked regions detected in the reference frame tracked using proposed method; (c) inpainted frames corresponding to frames in (b); (d) cracked regions detected independently in each frame; (e) inpainted frames corresponding to frames in (d).

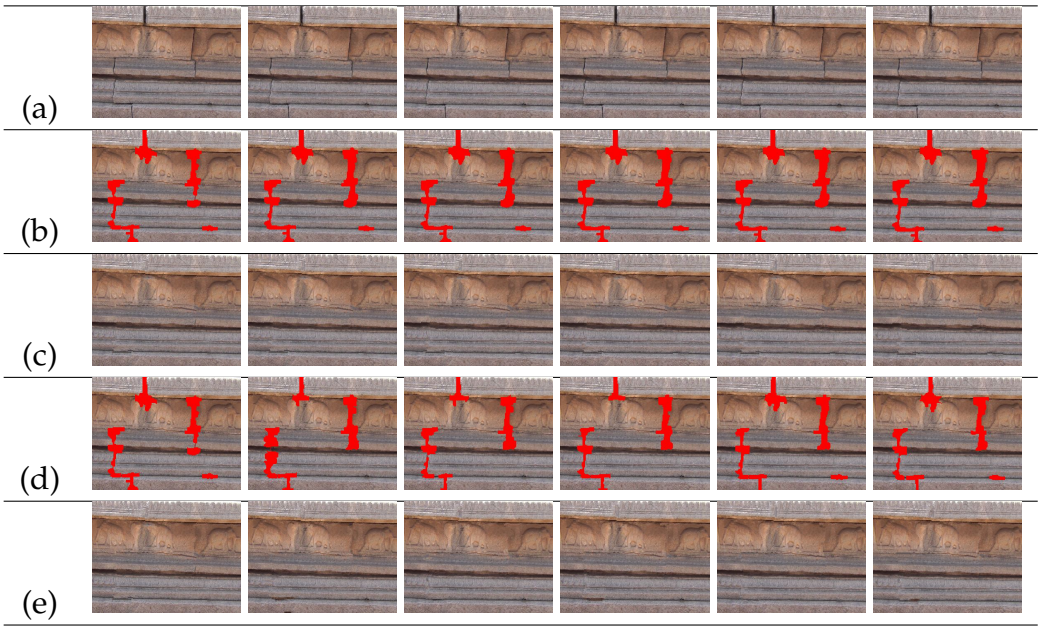


Figure 7.26: Result of auto-inpainting multiple cracked regions in video frames. (a) Input frame sequence, left most frame is the reference frame; (b) cracked regions detected in the reference frame tracked using proposed method; (c) inpainted frames corresponding to frames in (b); (d) cracked regions detected independently in each frame; (e) inpainted frames corresponding to frames in (d).

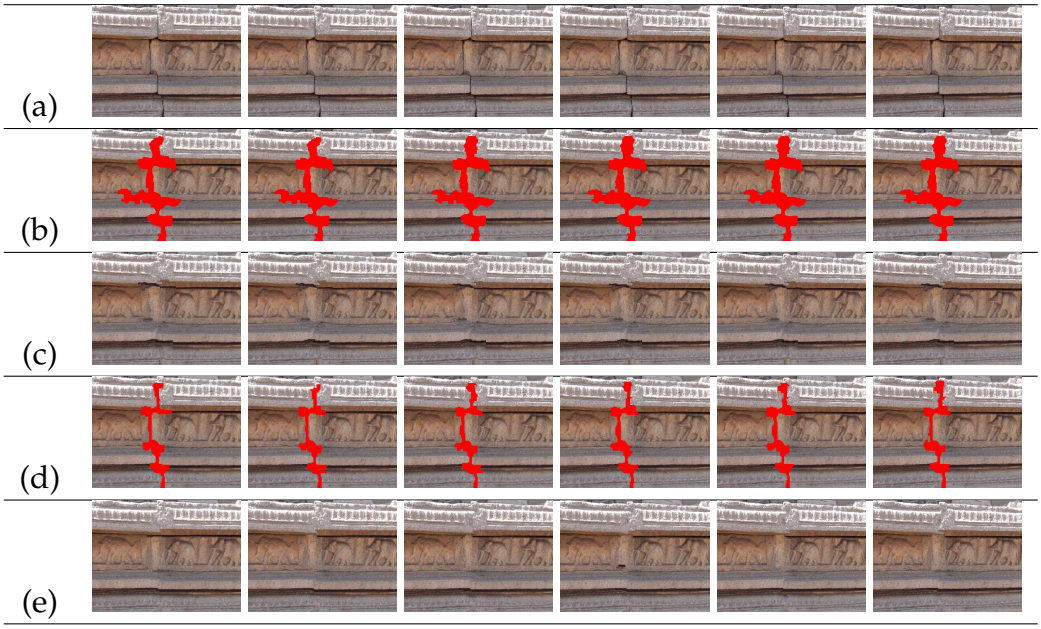


Figure 7.27: Result of auto-inpainting cracked regions in video frames containing artistic work. (a) Input frame sequence, left most frame is the reference frame; (b) cracked regions detected in the reference frame tracked using proposed method; (c) inpainted frames corresponding to frames in (b); (d) cracked regions detected independently in each frame; (e) inpainted frames corresponding to frames in (d).

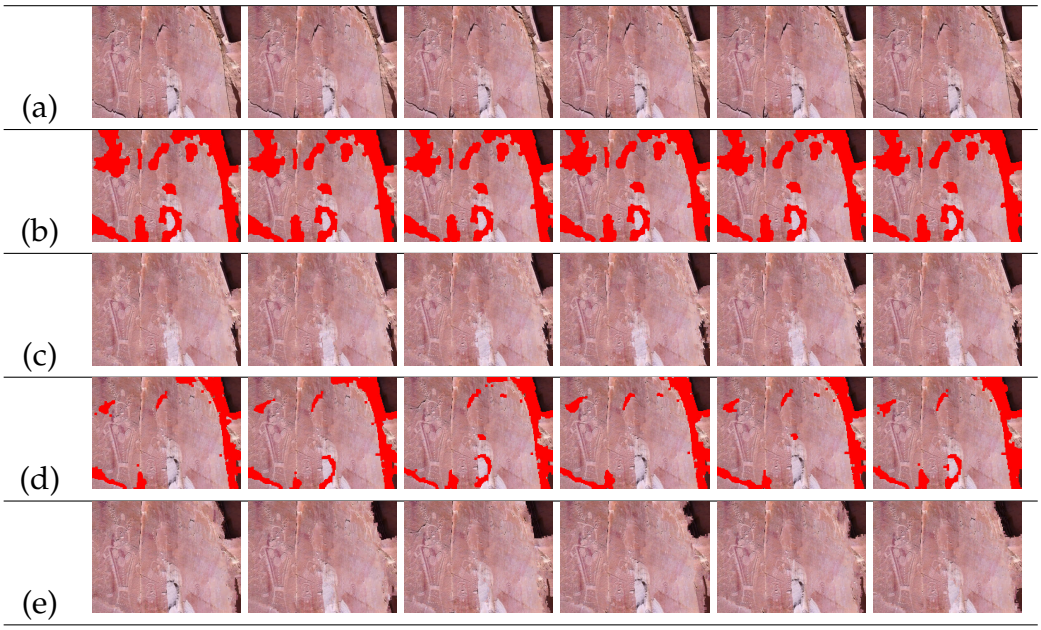


Figure 7.28: Result of auto-inpainting cracked regions in video frames captured at Utah containing petroglyphs. (a) Input frame sequence, left most frame is the reference frame; (b) cracked regions detected in the reference frame tracked using proposed method; (c) inpainted frames corresponding to frames in (b); (d) cracked regions detected independently in each frame; (e) inpainted frames corresponding to frames in (d).

seamless video. Although there exist approaches for semi-automatic inpainting of unwanted objects in videos [149] and video inpainting under constrained camera motion [111], it may be noted that, to the best of our knowledge, there does not exist any approach that demonstrates automatic video inpainting under unconstrained camera motion with no moving objects. Our approach handles these cases and we, therefore, do not show any comparison with the approaches in [111, 149]. However, we do compare the proposed approach with auto-inpainting done in a frame-by-frame fashion. The results of our method, along with auto-inpainting performed individually on every frame are shown in figures 7.24–7.28.

We now present an objective comparison of our method with frame-by-frame auto-inpainting in table 7.2 in terms of the NR VQA measures viz. blockiness, blurriness [38] and sudden local change [122, 123], along with the temporal consistency measure discussed in section 7.2.4. A video with higher blockiness and blur has higher value of the blockiness and blurriness metrics, respectively. For a temporally plausible video, the sudden local change is less while the temporal consistency measure has a higher value. From table 7.2 we observe that the proposed method performs better in terms of blockiness, sudden local change and temporal consistency, which is in accordance with the results in figures 7.24–7.28.

The implementation details along with the timing information are presented as follows. The cracked region detection in reference frames and independent inpainting of newly detected cracked pixels are implemented in Matlab. For frames

	Proposed method					**Frame-by-frame auto-inpainting**				
Video	**A**	**B**	**C**	**D**	**E**	**A**	**B**	**C**	**D**	**E**
Video1 (figure 7.24)	0.1125	5.1020	1.0737	0.9529	0.7501	0.1296	5.1073	1.3126	0.5064	0.2496
Video2 (figure 7.25)	0.1034	4.1261	1.5459	0.6671	0.9604	0.1270	4.2057	1.9463	0.1978	0.4148
Video3 (figure 7.26)	0.2975	4.3382	1.2908	0.9979	0.5424	0.2292	4.3666	1.5322	0.1862	0.6134
Video4 (figure 7.27)	0.1453	4.6306	1.8454	0.8173	0.9678	0.1473	4.7223	2.0858	0.2009	0.8946
Video4 (figure 7.28)	0.1582	3.1264	2.0559	0.5821	0.9654	0.1662	3.1586	2.7768	0.2301	0.9381

Table 7.2: Comparison of proposed method with frame-by-frame auto-inpainting, in terms of blockiness (**A**), bluriness (**B**), sudden local change (**C**) and temporal consistency in optical flow's direction (**D**) and magnitude (**E**).

of size 270×360 (for example the video corresponding to figure 7.24), the inpainting of reference frames takes nearly 1.5–2 seconds. This includes the time required for tracking and inpainting from previously detected cracked regions (about 0.6 seconds) followed by initial detection, refinement and inpainting of the newly detected cracked pixels. The first frame, however, required about 4.5 seconds for initial detection, refinement and inpainting. Note that the size of the frame here is 270×360. Subsequent (non-reference) frames take about 0.08 seconds to complete tracking and inpainting from previously detected cracked regions, which is very fast when compared to independent inpainting of each frame. In this case, a considerable amount of time is required for the inpainting operation in reference frames which introduces a lag in the video.

Although major computational steps are implemented in C (Matlab MEX), our implementation is not an optimized version but a proof of concept of the method discussed here. Having said that, we are optimistic about the implementation of our method for mobile phones in order to use the method directly on videos captured onsite. This is because of the quick inpainting of subsequent (non-reference) frames. A real-time on-the-fly inpainting of the video frames could be possible with an implementation optimized for the hardware of mobile phones.

7.3 Conclusion

In this chapter, we have first presented yet another technique that can automatically detect cracked regions and use these regions for inpainting. By comparing non-overlapping patches using the tolerant edit distance measure introduced here, our method initially localizes the cracked regions. Further, using an active contour-based segmentation, the results are refined to accurately detect the cracked regions. Thus we have progressed from chapter 6 where we discussed a primitive pixel based approach for crack detection approach in section 6.1 and a patch based method in section 6.2, to a sophisticated patch based approach discussed in this chapter, each showing improvement in the accuracy of detecting cracked regions. We have then discussed the extension of this approach to per-

form automatic inpainting of video frames. Here, every incoming frame is tested for being a reference frame wherein the newly appearing cracked pixels are inpainted independently. For subsequent non-reference frames, the tracking and inpainting of the cracked regions is performed by making use of the scale invariant feature transform (SIFT) and estimating homography between two temporally adjacent frames. In the end, we have also provided the temporal consistency measure to quantify the quality of the inpainted video. Reported results suggest the efficacy of our method to auto-inpaint the cracked regions in videos captured at heritage sites.

CHAPTER 8

Conclusions and Future Research Directions

In this chapter, we provide the conclusions by summarizing our main contributions and also indicate future research directions.

8.1 Conclusions

In this thesis we have addressed the problem of auto-inpainting by providing novel techniques for inpainting, simultaneous inpainting and super-resolution, followed by methods to automatically detect the regions to be inpainted considering the domain of digital heritage reconstruction. We began with a discussion of an exemplar based inpainting technique wherein we have used the pixel-neighborhood relationships as a cue for filling the missing pixels. This was done by estimating parameters of an AR model representing the pixel-neighborhood relationships. Unlike those approaches which simply copy the pixels to be inpainted from the best matching exemplar, we have used the estimated AR parameters in addition to the best matching exemplar to perform a seamless inpainting. We then proceeded towards discussing our second inpainting approach that not only inpaints the given image but also creates the missing details in its higher resolution i.e. it performs super-resolution inpainting. Simultaneously super-resolved regions in our proposed method are comparable to super-resolution of the inpainted regions obtained using the recent approach proposed in [51], and show greater details than those obtained using upsampling of the inpainted regions using bicubic interpolation.

In our next work, we have proposed a method that automates the process of

identifying the damage to visually dominant regions viz. eyes, nose and lips in face image of statues at a historic site, for the purpose of digital repair using inpainting. The dominant regions are extracted by considering the bilateral symmetry of face. Texton features are then used to identify the damaged regions. Poisson image editing method is then used to inpaint the damaged regions using a source region either from the same image or from other images depending on the extent of damage.

In our later works, we have introduced novel techniques that can automatically detect the cracked regions in heritage monuments and demonstrate their repair by inpainting. This can be particularly useful for performing on-the-fly digital reconstruction of damaged regions when tourists capture the heritage monuments using their handheld video capturing devices. Here, we have first discussed a simple pixel based method by making use of order-statistics and density filters. This is followed by a patch based technique that makes use of SVD for automatic detection of the cracked regions. This method shows improvement in the accuracy of detection in comparison to the simple pixel based approach. Following these two approaches, we discuss yet another effective and more accurate crack detection method based on comparison of non-overlapping patches using the novel tolerant edit distance measure, which is derived from a popular edit distance string metric used in the area of text mining. Thus, we have shown the progress from a primitive method to a sophisticated technique for detecting cracks.

Our next work involved extending our crack detection approach to perform auto-inpainting of video frames by making use of the scale invariant feature transform (SIFT) and homography. One may note that while most video inpainting techniques can inpaint moving objects under constrained camera motion or inpainting of a user-selected object [49, 99, 111, 126], to the best of our knowledge, there does not exist any approach that demonstrates fully automatic video inpainting (that also estimates the regions to be inpainted) under unconstrained camera motion with no moving objects, which is addressed in work. At the end of this work, we have provided video quality measures to quantify the temporal

consistency of inpainted videos.

Following are the major observations drawn from our work.

- Seams visible in inpainted results that arise due to the direct copying of the pixels from exemplars, can be avoided by making use of AR parameters for estimating the missing pixel values.
- Matching patches at a finer resolution by creating image representative LR-HR patch pair dictionaries helps not only in finding better source for inpainting, but also helps in creating exemplars whenever good matches are not available. This assists in performing better inpainting with simultaneous super-resolution.
- Using bilateral symmetry of face as cue helps in detection of the dominant facial regions while the texton features extracted from these regions assist in labeling them as either damaged or non-damaged.
- The pixel based approach using order-statistics along with density based filters and the patch based approaches using SVD and tolerant edit distance, can all be effectively used for detection of cracked regions. Nevertheless, the patch based approaches detect the cracked regions more accurately.
- In videos, better inpainting is obtained by detecting and inpainting the cracked regions quasi-periodically based on motion rather than performing detection and inpainting in a frame-by-frame fashion.
- The quality of inpainted videos can be quantified in terms of temporal consistency by making use of the optical flow estimated from corresponding input videos.

8.2 Future Research Directions

This thesis has presented novel approaches for inpainting and detection of damaged regions. In the process of this work, however, we identified related problems that one may consider worth pursuing. These are briefly described as follows.

- *Benchmarks for inpainting quality assessment*:

  We have seen in chapter 3 that, assigning subjective scores with the help of human observers is so far the most reliable way to quantify the quality of inpainted images, as well as for comparing the results of different inpainting methods. Apart from being subjective, such a comparison involves time-consuming exercises. Yet, one may note that the inpainting community continues to rely on subjective comparison of the inpainted results. There is, however, a pressing need to have objective measures that can be reliably used for comparing the results of various inpainting algorithms.

- *Real-time inpainting*:

  In chapter 7, we observed that although the detection of cracked regions and their propagation across subsequent frames was fast, a considerable amount of time was spent on inpainting the detected cracked regions. In general, the exemplar based inpainting methods perform several patch comparisons due to which the speed of inpainting is slow. Although one can consider the use of PatchMatch algorithm discussed in [7] to overcome this issue, it may be noted that the best source patches may not be selected due to a random search for matching patches. For enabling the real-time implementation of our auto-inpainting techniques, there is a need to have a faster inpainting method.

- *Inpainting based on semantics*:

  The existing inpainting techniques have been dependent on exemplars obtained by estimating patch similarity using a distance metric. However, none of these approaches use semantic information when performing inpainting. We human beings have developed a great ability to identify various objects even if they are partially occluded. Not only that, we can also "imagine the completed object" from whatever is visible to us. Thus, we humans use a two step process: (1) use semantics to group visual information into meaningful objects and (2) if occluded, use knowledge from our past experiences to visualize the identified object in its complete form. This observation provides the motivation to explore the image inpainting problem in

a machine-learning perspective where the system needs to be trained prior to performing the desired task. For example consider a scenario wherein an occluded chair is to be completed using inpainting. Here, without the use of semantic information, it is likely that the inpainted regions contain extended edges of the chair or a plain background. One can intuitively expect a better result if the inpainting is done with knowledge of a prior information that the occluded object is a chair and the exemplar search can then be restricted to chairs. To the best of our knowledge, the removal of large missing regions has not been addressed using machine learning approaches, which is worth pursuing. Nevertheless, works this direction have been very recently proposed in [110, 119] or available as an archived manuscript in [157].

References

[1] M. V. Afonso and J. M. R. Sanches. Blind inpainting using l_0 and total variation regularization. *IEEE Trans. Image Processing*, 24(7):2239–2253, July 2015.

[2] S. Al-Takrouri and A. V. Savkin. A model validation approach to texture recognition and inpainting. *Pattern Recognition*, 43(6):2054–2067, 2010.

[3] L. Alvarez, P.-L. Lions, and J.-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal Numerical Analysis*, 29:845–866, June 1992.

[4] T. Amano. Correlation based image defect detection. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 01 of *ICPR '06*, pages 163–166, Washington, DC, USA, 2006. IEEE Computer Society.

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

[6] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *IEEE Trans. Pattern Analy. Machine Intell.*, 24(9):1167–1183, Sept. 2002.

[7] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graphics*, 28(3):24:1–24:11, July 2009.

[8] N. Behzad and D. Qarizadah. The man who helped blow up the bamiyan buddhas. *BBC News, Asia*, 12 March 2015 2015. URL http://www.bbc.com/news/world-asia-31813681.

[9] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. 27th Annual Conf. Computer Graphics and Interactive Techniques*, pages 417–424, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[10] A. V. Bhavsar and A. Rajagopalan. Inpainting large missing regions in range images. In *Pattern Recognition, International Conference on*, pages 3464–3467, 2010.

[11] A. V. Bhavsar and A. N. Rajagopalan. Range map with missing data - joint resolution enhancement and inpainting. In *Indian Conference on Computer Vision, Graphics & Image Processing*, pages 359–365, 2008.

[12] A. V. Bhavsar and A. N. Rajagopalan. Inpainting in multi-image stereo. In *Proceedings of the 32Nd DAGM Conference on Pattern Recognition*, pages 172–181, 2010.

[13] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *Int J. Comput. Vision*, 74(1):59–73, Aug. 2007.

[14] A. Bugeau, M. Bertalmío, V. Caselles, and G. Sapiro. A comprehensive framework for image inpainting. *IEEE Trans. Image Processing*, 19(10):2634–2645, Oct. 2010.

[15] 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*, pages 1–13, 2015.

[16] E. J. Candes and J. Romberg. ℓ_1-magic - recovery of sparse signals via convex programming, 2005. URL http://users.ece.gatech.edu/~justin/l1magic/#code.

[17] E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, March 2008.

[18] R. Chan, T. Chan, Z. Shen, T. F. Chan, M. K. Ng, A. C. Yau, and A. M. Yip. Superresolution image reconstruction using fast inpainting algorithms. *Applied and Computational Harmonic Analysis, Special Issue on Mathematical Imaging*, 23(1):3–24, 2007.

[19] T. Chan and L. Vese. Active contours without edges. *IEEE Trans. Image Processing*, 10(2):266–277, Feb. 2001.

[20] R.-C. Chang, Y.-L. Sie, S.-M. Chou, and T. K. Shih. Photo defect detection for image inpainting. In *Proceedings of the Seventh IEEE International Symposium on Multimedia*, ISM '05, pages 403–407, Washington, DC, USA, 2005. IEEE Computer Society.

[21] S. Chaudhuri. *Super-Resolution Imaging*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.

[22] S. Chaudhuri and M. V. Joshi. *Motion-Free Super-Resolution*. Springer US, 2005. ISBN 9780387258904.

[23] D. Chen and R. J. Plemmons. *The birth of numerical analysis*, chapter Nonnegativity Constraints in Numerical Analysis, pages 109–138. World Scientific, 2009.

[24] B. Cornelis, T. RuÅ¿iÄĞ, E. Gezels, A. Dooms, A. PiÅ¿urica, L. PlatiÅąa, J. Cornelis, M. Martens, M. D. Mey, and I. Daubechies. Crack detection and inpainting for virtual restoration of paintings: The case of the ghent altarpiece. *Signal Processing*, 93(3):605–619, 2013. Image Processing for Digital Art Work.

[25] A. Criminisi, P. Pérez, and K. Toyama. Object removal by exemplar-based inpainting. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 2, page 721, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[26] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. Image Processing*, 13:1200–1212, 2004.

[27] Z. Cui, H. Chang, S. Shan, B. Zhong, and X. Chen. Deep network cascade for image super-resolution. In *Computer Vision – ECCV 2014: 13th European Conference, Part V*, pages 49–64, 2014.

[28] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen. Image melding: Combining inconsistent images using patch-based synthesis. *ACM Trans. Graphics*, 31(4):82:1–82:10, July 2012.

[29] C. de Kadt, J. Gain, and P. Marais. Revisiting district six: a case study of digital heritage reconstruction from archival photographs. In *Proceedings of the 6th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, AFRIGRAPH '09, pages 13–21, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-428-7.

[30] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *J. American. Soc. for Information Sci.*, 41 (6):391–407, 1990.

[31] B. Dong, H. Ji, J. Li, Z. Shen, and Y. Xu. Wavelet frame based blind image inpainting. *Applied and Computational Harmonic Analysis*, 32(2):268 – 279, 2012.

[32] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *Computer Vision – ECCV 2014: 13th European Conference, Part IV*, pages 184–199, 2014.

[33] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Analy. Machine Intell.*, 38(2): 295–307, Feb 2016.

[34] S. F. El-Hakim. Semi-automatic 3d reconstruction of occluded and unmarked surfaces from widely separated views. In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Greece*, pages 143–148, 2002.

[35] S. F. El-Hakim, G. MacDonald, J.-F. Lapointe, L. Gonzo, and M. Jemtrud. On the digital reconstruction and interactive presentation of heritage sites through time. In *VAST 2006: The 7th International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage, Nicosia, Cyprus, 2006. Proceedings*, pages 243–250. Eurographics Association, 2006.

[36] M. Elad and M. Aharon. Image denoising via learned dictionaries and sparse representation. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 895–900, June 2006.

[37] C. Emerson, N. Lam, and D. Quattrochi. Multi-scale fractal analysis of image texture and patterns. *Photogrammetric Engg. and Remote Sensing*, 65(1): 51–62, Jan. 1999.

[38] M. Farias and S. Mitra. No-reference video quality metric based on artifact measurements. In *Proc. Int. Conf. Image Processing*, volume 3, pages III–141–4, 2005.

[39] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super-resolution. *IEEE Trans. Image Processing*, 13(10):1327–1344, Oct. 2004.

[40] R. Fattal. Image upsampling via imposed edge statistics. *ACM Trans. Graphics*, 26(3):95, July 2007.

[41] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. Technical Report RR-0856, INRIA, June 1988.

[42] X. Feng and P. Milanfar. Multiscale principal components analysis for image local orientation estimation. In *Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on*, volume 1, pages 478–482 vol.1, Nov 2002.

[43] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[44] G. Freedman and R. Fattal. Image and video upscaling from local self-examples. *ACM Trans. Graphics*, 28(3):1–10, Apr. 2011.

[45] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22(2):56–65, Mar. 2002.

[46] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Analy. Machine Intell.*, 32(8):1362–1376, Aug. 2010.

[47] P. P. Gajjar and M. V. Joshi. A fast approach to the learning based super-resolution using autoregressive model prior and wavelet prior. In *Proceedings of the 14th National Conference on Communications*, pages 229–233, 2008.

[48] P. P. Gajjar and M. V. Joshi. New learning based super-resolution: Use of DWT and IGMRF prior. *IEEE Trans. Image Processing*, 19(5):1201–1213, May 2010.

[49] M. Ghorai, P. Purkait, and B. Chanda. A fast video inpainting technique. In *Pattern Recognition and Machine Intelligence*, volume 8251 of *Lecture Notes in Computer Science*, pages 430–436. Springer Berlin Heidelberg, 2013.

[50] M. Ghorai, S. Mandal, and B. Chanda. Image completion assisted by transformation domain patch approximation. In *Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing*, ICVGIP '14, pages 66:1–66:8, New York, NY, USA, 2014. ACM.

[51] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *Proc. 12th IEEE Conf. Computer Vision*, pages 349–356, Mar. 2009.

[52] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 2001. ISBN 0201180758.

[53] Google Images, [Online] Available: http://www.images.google.com, Mar. 2012.

[54] H. Grossauer. A combined pde and texture synthesis approach to inpainting. In *8th European Conference on Computer Vision (ECCV 2004)*, volume 4, pages 214–224, 2004.

[55] A. Grün, F. Remondino, and L. Zhang. Photogrammetric reconstruction of the great buddha of bamiyan, afghanistan. *The Photogrammetric Record*, 19 (107):177–199, sept 2004.

[56] C. Guillemot and O. L. Meur. Image inpainting : Overview and recent advances. *IEEE Signal Processing Magazine*, 31(1):127–144, 2014.

[57] P. F. Harrison. Gimp resynthesizer plugin, 2011. URL http://www.logarithmic.net/pfh/resynthesizer.

[58] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[59] K. He and J. Sun. Statistics of patch offsets for image completion. In *12th European Conference on Computer Vision (ECCV 2012)*, pages 16–29, 2012.

[60] R. K. Hladky and S. D. Pauls. Minimal surfaces in the roto-translation group with applications to a neuro-biological image completion model. *Journal of Mathematical Imaging and Vision*, 36(1):1–27, Jan. 2010.

[61] X. Huan, B. Murali, and A. L. Ali. Image restoration based on the fast marching method and block based sampling. *Computer Vision and Image Understanding*, 114(8):847–856, Aug. 2010.

[62] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf. Image completion using planar structure guidance. *ACM Trans. Graphics*, 33(4):129:1–129:10, July 2014.

[63] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf. Dataset for image completion using planar structure guidance, 2014. URL https://sites.google.com/site/jbhuang0604/publications/struct_completion.

[64] J.-B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 2015.

[65] T. S. Huang and R. Y. Tsai. Multi-frame image restoration and registration. *Advances in Computer Vision and Image Processing*, 1:317–339, 1984.

[66] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP: Graphical Model and Image Processing*, 53(3):231–239, Apr. 1991.

[67] S. Jeannin and A. Divakaran. MPEG-7 visual motion descriptors. *IEEE Trans. Circuits and Syst. for Video Tech.*, 11(6):720–724, 2001.

[68] D. J. Jobson, Z. Rahman, and G. A. Woodell. Properties and performance of a center/surround retinex. *IEEE Trans. Image Processing*, 6(3):451–462, Mar 1997.

[69] M. V. Joshi, L. Bruzzone, and S. Chaudhuri. A model-based approach to multiresolution fusion in remotely sensed images. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(9):2549–2562, Sept. 2006.

[70] S. Katahara and M. Aoki. Face parts extraction windows based on bilateral symmetry of gradient direction. In *Computer Analysis of Images and Patterns*, volume 1689, pages 834–834. Springer Berlin Heidelberg, 1999.

[71] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[72] J. F. Kenney. *Mathematicals of Statistics*. Van Nostrand, 1954.

[73] N. Khatri and M. V. Joshi. Image super-resolution: Use of self-learning and gabor prior. In *11th Asian Conference on Computer Vision (ACCV 2012)*, pages 413–424, 2012.

[74] N. Khatri and M. V. Joshi. Efficient self-learning for single image upsampling. In *22nd International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 2014)*, pages 1–8, 2014.

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

[76] M. Köppel, M. B. Makhlouf, K. MÃijller, and T. Wiegand. Fast image completion method using patch offset statistics. In *Proc. Int. Conf. Image Processing*, pages 1795–1799, Sept 2015.

[77] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia, 2005. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns /robust/ransacfithomography.m>.

[78] M. Kulkarni and A. N. Rajagopalan. Depth inpainting by tensor voting. *Journal of the Optical Society of America A*, 30(6):1155–1165, Jun 2013.

[79] C. Kwan, B. Ayhan, G. Chen, J. Wang, B. Ji, and C.-I. Chang. A novel approach for spectral unmixing, classification, and concentration estimation of chemical and biological agents. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(2):409–419, Feb. 2006.

[80] T.-H. Kwok, H. Sheung, and C. C. L. Wang. Fast query for exemplar-based image completion. *IEEE Trans. Image Processing*, 19(12):3106–3115, Dec. 2010.

[81] O. Le Meur and C. Guillemot. Super-resolution-based inpainting. In *12th European Conference on Computer Vision (ECCV 2012)*, volume 7577 of *Lecture Notes in Computer Science*, pages 554–567. Springer Berlin Heidelberg, 2012.

[82] Y. Lee and A. Fam. An edge gradient enhancing adaptive order statistic filter. *IEEE Trans. Acoustics, Speech & Signal Proc.*, 35(5):680–695, May 1987.

[83] Y.-H. Lee and S. Tantaratana. An adaptive edge enhancing order statistic filter. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, volume 3, pages 1518–1521, Apr. 1988.

[84] P. Legrand. Local regularity and multifractal methods for image and signal analysis. In *Scaling, Fractals and Wavelets*. Wiley, Jan. 2009.

[85] H. Li, S. Wang, W. Zhang, and M. Wu. Image inpainting based on scene transform and color transfer. *Pattern Recognition Letters*, 31(7):582–592, May 2010.

[86] Z. Lin and H.-Y. Shum. Fundamental limits of reconstruction-based super-resolution algorithms under local translation. *IEEE Trans. Pattern Analy. Machine Intell.*, 26(1):83–97, Jan. 2004.

[87] D. Liu, Z. Wang, B. Wen, J. Yang, W. Han, and T. S. Huang. Robust single image super-resolution via deep networks with sparse prior. *IEEE Trans. Image Processing*, 25(7):3194–3207, July 2016.

[88] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[89] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. 7th Int. Joint Conf. on AI*, pages 674–679, 1981.

[90] Q. Luong, T. Ruzic, A. Pizurica, and W. Philips. Single-image super-resolution using sparsity constraints and non-local similarities at multiple resolution scales. In *Proceedings of the Scoiety Of Photo-Optical Instrumentation Engineers (SPIE)*, volume 7723, page 8, 2010.

[91] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003. ISBN 0387008934.

[92] L. W. MacDonald. *Digital Heritage: Applying Digital Imaging to Cultural Heritage*. Elsevier, 2006. ISBN 9780750661836.

[93] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Trans. Image Processing*, 17(1):53–69, Jan 2008.

[94] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proc. 12th IEEE Conf. Computer Vision*, pages 2272–2279, Sept 2009.

[95] S. Masnou and J.-M. Morel. Level lines based disocclusion. In *Proc. Int. Conf. Image Processing*, volume 3, pages 259–263, Oct. 1998.

[96] S. McCloskey, M. Langer, and K. Siddiqi. Removal of partial occlusion from single images. *IEEE Trans. Pattern Analy. Machine Intell.*, 33(3):647–654, march 2011. doi: 10.1109/TPAMI.2010.187.

[97] A. Mittal, A. K. Moorthy, and A. C. Bovik. No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Processing*, 21(12):4695–4708, Dec. 2012.

[98] U. M. Munshi and B. B. Chaudhuri. *Multimedia Information Extraction and Digital Heritage Preservation*. Platinum Jubilee series. World Scientific, 2011. ISBN 9789814307253.

[99] A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Pérez. Video inpainting of complex scenes. *SIAM Journal on Imaging Sciences, Society for Industrial and Applied Mathematics*, 7(4):1993–2019, 2014.

[100] M. M. Oliveira, B. Bowen, R. Mckenna, and Y. sung Chang. Fast digital image inpainting. In *Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001)*, pages 261–266. ACTA Press, 2001.

[101] M. G. Padalkar and M. V. Joshi. *Automatic Detection and Inpainting of Defaced Regions and Cracks in Heritage Monuments*. [**Revised draft submitted** to Prof. Santanu Chaudhury, IIT Delhi whose proposal for publishing an edited volume of chapters outlining the outcome of different aspects of the Indian Digital Heritage (IDH)-Hampi project has been approved by Springer-Verlag (http://digitalhampi.in/#book)].

[102] M. G. Padalkar and M. V. Joshi. Auto-inpainting heritage scenes: a complete framework for detecting and infilling cracks in images and videos with quantitative assessment. *Machine Vision and Applications*, 26(2):317–337, 2015.

[103] M. G. Padalkar, M. V. Joshi, M. A. Zaveri, and C. M. Parmar. Exemplar based inpainting using autoregressive parameter estimation. In *Proceedings of the International Conference on Signal, Image and Video Processing*, ICSIVP'12, pages 154–160, IIT Patna, India, Jan. 2012.

[104] M. G. Padalkar, M. V. Vora, M. V. Joshi, M. A. Zaveri, and M. S. Raval. Identifying vandalized regions in facial images of statues for inpainting. In *New Trends in Image Analysis and Processing–ICIAP 2013*, volume 8158 of *Lecture Notes in Computer Science*, pages 208–217. Springer Berlin Heidelberg, Sept. 2013.

[105] M. G. Padalkar, M. A. Zaveri, and M. V. Joshi. SVD based automatic detection of target regions for image inpainting. In J.-I. Park and J. Kim, editors, *Computer Vision - ACCV 2012 Workshops*, volume 7729 of *Lecture Notes in Computer Science*, pages 61–71. Springer Berlin Heidelberg, 2013.

[106] M. G. Padalkar, M. V. Joshi, and N. Khatri. Simultaneous inpainting and super-resolution using self-learning. In X. Xie, M. W. Jones, and G. K. L. Tam, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 105.1–105.12. BMVA Press, September 2015.

[107] M. G. Padalkar, M. V. Joshi, and N. L. Khatri. *Digital Heritage Reconstruction Using Super-resolution and Inpainting*. Synthesis Lectures on Visual Computing, Morgan & Claypool Publishers, December 2016.

[108] S. C. Park, M. K. Park, and M. G. Kang. Super-resolution image reconstruction: a technical overview. *Signal Processing Magazine, IEEE*, 20(3):21–36, May 2003.

[109] C. M. Parmar, M. V. Joshi, M. S. Raval, and M. A. Zaveri. Automatic image

inpainting for the facial images of monuments. In *Proceedings of Electrical Engineering Centenary Conference 2011*, pages 415–420, 14–17 December 2011.

[110] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 2016. (accepted).

[111] K. A. Patwardhan, G. Sapiro, and M. Bertalmío. Video inpainting under constrained camera motion. *IEEE Trans. Image Processing*, 16(2):545–553, 2007.

[112] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *Proc. 30th Annual Conf. Computer Graphics and Interactive Techniques*, pages 313–318, New York, NY, USA, 2003. ACM.

[113] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. Graphics*, 22(3):313–318, July 2003.

[114] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Analy. Machine Intell.*, 12:629–639, July 1990.

[115] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *Proc. 12th IEEE Conf. Computer Vision*, pages 151–158, Sept 2009.

[116] P. Purkait and B. Chanda. Digital restoration of damaged mural images. In *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, ICVGIP '12, pages 49:1–49:8, New York, NY, USA, 2012. ACM.

[117] P. Purkait and B. Chanda. Super resolution image reconstruction through bregman iteration using morphologic regularization. *IEEE Trans. Image Processing*, 21(9):4029 –4039, sept. 2012.

[118] K. Rematas, T. Ritschel, M. Fritz, and T. Tuytelaars. Image-based synthesis and re-synthesis of viewpoints guided by 3d models. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 3898–3905, June 2014.

[119] K. Rematas, C. Nguyen, M. Fritz, and T. Tuytelaars. Novel views of objects from a single image. *IEEE Trans. Pattern Analy. Machine Intell.*, 2016. (to appear).

[120] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graphics*, 23(3):309–314, Aug. 2004.

[121] D. L. Ruderman and W. Bialek. Statistics of natural images: Scaling in the woods. *Physical Review Letters*, 73(6):814–817, Aug. 1994.

[122] M. Saad, A. Bovik, and C. Charrier. Blind prediction of natural video quality. *IEEE Trans. Image Processing*, 23(3):1352–1365, Mar. 2014.

[123] M. A. Saad and A. C. Bovik. Blind quality assessment of videos using a model of natural scene statistics and motion coherency. In *Asilomar Conference on Signals, Systems, and Computers*, pages 332–336, Nov. 2012.

[124] S. Scholes. Mcconkie ranch petroglyphs near vernal, utah, Nov. 2011. URL https://www.youtube.com/watch?v=jmewuqEXTK8. [Accessed 01 Sept. 2014].

[125] T. Shibata, A. Iketani, and S. Senda. Image inpainting based on probabilistic structure estimation. In *10th Asian Conference on Computer Vision (ACCV 2010)*, volume III, pages 109–120, Berlin, Heidelberg, 2011. Springer-Verlag.

[126] T. Shih, N. Tang, and J.-N. Hwang. Exemplar-based video inpainting without ghost shadow artifacts by maintaining temporal continuity. *IEEE Trans. Circuits and Syst. for Video Tech.*, 19(3):347–360, Mar. 2009.

[127] A. B. Siddiqui, M. Rashid, M. A. Jaffar, A. Hussain, and A. M. Mirza. Feature classification for multi-focus image fusion. *International Journal of the Physical Sciences*, 6(20):4838–4847, Sept. 2011.

[128] T. Sikora. MPEG digital video-coding standards. *Signal Processing Magazine, IEEE*, 14(5):82–100, 1997.

[129] F. Stanco, S. Battiato, and G. Gallo. *Digital Imaging for Cultural Heritage Preservation: Analysis, Restoration, and Reconstruction of Ancient Artworks*. Digital Imaging and Computer Vision. CRC Press, 2011. ISBN 9781439821749.

[130] G. R. K. S. Subrahmanyam, A. N. Rajagopalan, and R. Aravind. Recursive framework for joint inpainting and de-noising of photographic films. *Journal of the Optical Society of America A*, 27(5):1091–1099, May 2010.

[131] J. Sun, Z. Xu, and H.-Y. Shum. Image super-resolution using gradient profile prior. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[132] H. Takeda, H. J. Seo, and P. Milanfar. Statistical approaches to quality assessment for image restoration. In *Consumer Electronics, 2008. ICCE 2008. Digest of Technical Papers. International Conference on*, pages 1–2, Jan. 2008. doi: 10.1109/ICCE.2008.4587957.

[133] T. Tamaki, H. Suzuki, and M. Yamamoto. String-like occluding region extraction for background restoration. *International Conference on Pattern Recognition*, 3:615–618, 2006. ISSN 1051-4651.

[134] C. V. L. The University of Tokyo. Virtual Asukakyo, 2013. URL http://www.cvl.iis.u-tokyo.ac.jp/research/virtual-asukakyo/.

[135] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of Royal Stat. Soc., B*, 63(2):411–423, 2001.

[136] R. Timofte, V. De, and L. V. Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Proc. 15th IEEE Conf. Computer Vision*, pages 1920–1927, Dec 2013.

[137] R. Timofte, , V. De Smet, and L. Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *12th Asian Conference on Computer Vision (ACCV 2014)*, Nov. 2014.

[138] D. T. Trung, A. Beghdadi, and M.-C. Larabi. Perceptual evaluation of digital image completion quality. In *21st European Signal Processing Conference (EUSIPCO 2013)*, pages 1–5, Sept 2013.

[139] D. T. Trung, A. Beghdadi, and M.-C. Larabi. Perceptual quality assessment for color image inpainting. In *Proc. Int. Conf. Image Processing*, pages 398–402, Sept 2013.

[140] N. Turakhia, R. Shah, and M. Joshi. Automatic crack detection in heritage site images for image inpainting. In *Eighth Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, page 68, 2012.

[141] A. Turiel, G. Mato, N. Parga, and J. pierre Nadal. The self-similarity properties of natural images resemble those of turbulent flows. *Physical Review Letters*, 80:1098–1101, Feb. 1998.

[142] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *7th European Conference on Computer Vision (ECCV 2002)*, pages 255–271, 2002.

[143] V. Štruc and N. Pavešić. *Photometric normalization techniques for illumination invariance*, pages 279–300. IGI-Global, 2011.

[144] R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, Jan. 1974.

[145] M. E. Wall, A. Rechtsteiner, and L. M. Rocha. *Singular value decomposition and principal component analysis*, chapter Singular value decomposition and principal component analysis, pages 91–109. Kluwer: Norwell, 2003.

[146] Y. Wang, A. Szlam, and G. Lerman. Robust locally linear analysis with applications to image denoising and blind inpainting. *SIAM Journal on Imaging Sciences*, 6(1):526–562, 2013.

[147] Z. Wang, F. Zhou, and F. Qi. Inpainting thick image regions using isophote propagation. In *Proc. Int. Conf. Image Processing*, pages 689–692, Oct. 2006.

[148] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. In *Proc. 17th IEEE Conf. Computer Vision*, pages 370–378, Dec 2015.

[149] Y. Wexler, E. Shechtman, and M. Irani. Space-time completion of video. *IEEE Trans. Pattern Analy. Machine Intell.*, 29(3):463–476, March 2007.

[150] O. Whyte, J. Sivic, and A. Zisserman. Get out of my picture! internet-based inpainting. In *Proceedings of the 20th British Machine Vision Conference, London*, 2009.

[151] J. Wu and Q. Ruan. Object removal by cross isophotes exemplar-based inpainting. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 810–813, Washington, DC, USA, 2006. IEEE Computer Society.

[152] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Proc. Systems 25, Proc. 26nd Annual Conf. on Neural Information Proc. Systems*, pages 350–358, 2012.

[153] Z. Xiong, X. Sun, and F. Wu. Block-based image compression with parameter-assistant inpainting. *IEEE Trans. Image Processing*, 19(6):1651–1657, June 2010.

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

[155] M. Yan. Restoration of images corrupted by impulse noise and mixed gaussian impulse noise using blind inpainting. *SIAM Journal on Imaging Sciences*, 6(3):1227–1245, 2013.

[156] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE Trans. Image Processing*, 19(11):2861–2873, Nov. 2010.

[157] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2016.

[158] Q. Yuan, L. Zhang, and H. Shen. Regional spatially adaptive total variation super-resolution with spatial information filtering and clustering. *IEEE Trans. Image Processing*, 22(6):2327–2342, June 2013.

[159] T. H. Zaveri and M. A. Zaveri. A novel region based multimodality image fusion method. *Journal of Pattern Recognition Research*, 6(2):140–153, February 2011.

[160] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang. Cracktree: Automatic crack detection from pavement images. *Pattern Recognition Letters*, 33(3):227 – 238, 2012.

List of Publications

Book

- M. G. Padalkar, M. V. Joshi and N. L. Khatri, *Digital Heritage Reconstruction Using Super-resolution and Inpainting*. Synthesis Lectures on Visual Computing, Morgan & Claypool Publishers, December 2016.

Book Chapter

- M. G. Padalkar and M. V. Joshi, "Automatic Detection and Inpainting of Defaced Regions and Cracks in Heritage Monuments," [**Revised draft submitted** to Prof. Santanu Chaudhury, IIT Delhi whose proposal for publishing an edited volume of chapters outlining the outcome of different aspects of the Indian Digital Heritage (IDH)-Hampi project has been approved by Springer-Verlag (http://digitalhampi.in/#book)].

Journal

- M. G. Padalkar and M. V. Joshi, "Auto-inpainting Heritage Scenes: A Complete Framework for Detecting and Infilling Cracks in Images and Videos with Quantitative Assessment," in *Machine Vision and Applications (MVA)*, vol. 26, no. 2–3, 317–337, March 2015.

International Conference Proceedings

- M. G. Padalkar, M. V. Joshi and N. Khatri, "Simultaneous Inpainting and Super-resolution using Self-learning," in *26th British Machine Vision Conference (BMVC 2015)*, Swansea, UK, Sept. 7-10, 2015, pp. 105.1–105.12.

- M. G. Padalkar, M. V. Vora, M. V. Joshi, M. A. Zaveri, and M. S. Raval, "Identifying vandalized regions in facial images of statues for inpainting," in *New Trends in Image Analysis and Processing–ICIAP 2013*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, Sept. 2013, vol. 8158, pp. 208–217.

- M. G. Padalkar, M. A. Zaveri, and M. V. Joshi, "SVD based automatic detection of target regions for image inpainting," in *Computer Vision – ACCV 2012 Workshops*, ser. Lecture Notes in Computer Science, J.-I. Park and J. Kim, Eds., vol. 7729. Springer Berlin Heidelberg, 2013, pp. 61–71.

- M. G. Padalkar, M. V. Joshi, M. A. Zaveri, and C. M. Parmar, "Exemplar based inpainting using autoregressive parameter estimation," in *Proceedings of the International Conference on Signal, Image and Video Processing (ICSIVP'12)*, IIT Patna, India, 2012, pp. 154–160.

Achievements

- **September 2015**

  Xerox Research Centre India (XRCI) Travel Grant of INR 1,25,000.00 for presenting paper in the 26^{th} British Machine Vision Conference (BMVC 2015) at Swansea, United Kingdom.

- **April–November 2014**

  Organizing Committee Member in the 3rd ACCV Workshop on e-Heritage (Held conjunction with the 12th Asian Conference on Computer Vision – ACCV 2014).

- **September 2014**

  Reviewer in the Eighth International Conference on Advances in Pattern Recognition (ICAPR 2015)
  (Invited by: Program Chairs, ICAPR 2015).

- **January 2014**

  Reviewer in Multimedia Tools and Applications
  (Invited by: The Editorial Office, Multimedia Tools and Applications).