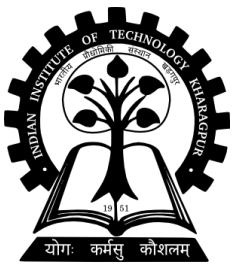
Image De-fencing using Microsoft Kinect

Vikram Voleti

09EE3501



Department of Electrical Engineering

Indian Institute of Technology Kharagpur

May 2014

Image De-fencing using Microsoft Kinect

Thesis submitted to Indian Institute of Technology Kharagpur
in partial fulfillment of the requirements for the award of the degree of

Master of Technology

in Electrical Engineering

with specialization in

Instrumentation and Signal Processing

by

Vikram Voleti

under the guidance of

Rajiv Ranjan Sahay

Department of Electrical Engineering

Indian Institute of Technology Kharagpur

May 2014

Declaration of Authorship

I, VIKRAM VOLETI, declare that this thesis titled, "Image De-fencing using Microsoft Kinect" and the work presented in it are my own. I confirm that:

- The work contained in this report is original and has been done by me under the guidance of my supervisor.
- This work was done wholly or mainly while in candidature for a degree at IIT Kharagpur.
- The work has not been submitted for a degree or any other qualification in any other institution.
- Where I have consulted the published work of others, this is always clearly attributed.
- I have acknowledged all main sources of help.
- I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

_____

Vikram Voleti [09EE3501]

April 29th, 2014

IIT Kharagpur

Certificate

This is to certify that the report entitled, "Image De-fencing using Microsoft Kinect" submitted to the Indian Institute of Technology Kharagpur, India, for the award of the degree of Master of Technology in Electrical Engineering by Mr. "VIKRAM VOLETI" is a record of bonafide research work carried out by him under my supervision and guidance. The thesis has reached the standard fulfilling the requirements of the regulations related to the degree. The results embodied in the thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

_____

Rajiv Ranjan Sahay
Department of Electrical Engineering
IIT Kharagpur

Acknowledgements

I would like to gratefully and sincerely express my deep sense of gratitude to my project supervisor Dr. Rajiv Sahay for his unwavering support, insightful suggestions, encouragement and fruitful discussions throughout the duration of my project. He has always made himself available for every sort of discussion. I have learned a great deal in academic areas as well as matters of one's attitude towards life over the past two years through his uninhibited help, motivation and support. It has been a privilege working with him over a project that he was as dedicated to as much as I was, if not more. It goes to his credit that as a student I have been given the opportunity to develop my own individuality and self-sufficiency by being allowed to work with such independence.

I would also like to thank Ganesh Jonna for his immense contribution in getting this project to fruition. He always by my side during experimentation and development of the results, and his constant presence inspired me to achieve more than what I would have achieved working alone. I would also like to express my gratitude to my fellow batchmates of Electrical Engineering, in particular Koustav Brahma for his timely help and technical support.

I am also grateful to all the faculty members, technical staff and research students of the Department of Electrical Engineering and School of Information Technology for their help, suggestions and comments throughout the tenure of the work. I would also like to express my gratitude to the School of Medical Science and Technology at IIT Kharagpur, for all the facilities that were unwittingly provided which proved crucial in developing this project.

Finally, I would like to express my thanks and love to my parents for all their love, support and encouragement. None of this would have been possible without them.

Vikram Voleti [09EE3501]
April 29th, 2014

Image De-fencing using Microsoft Kinect

by

Vikram Voleti

Submitted to the Indian Institute of Technology Kharagpur in April, 2014 in partial fulfillment of the requirements for the degree of Master of Technology in Electrical Engineering.

Abstract

De-fencing of images has been a problem that has been tackled in several different ways, mathematically and indirectly. The advent of depth sensors such as the Microsoft Kinect offers a new way of solving this problem. In this project, a multimodal approach to de-fencing of images has been explored, by recording data in the colour (RGB) domain as well as depth (D) domain, thus making RGB-D data. Depth data has been effectively used to detect fence locations.

In order to in-paint the regions in the image occluded by the fence, multiple views of the same scene were captured so that pixel information can be suitably borrowed from these different frames. But it is necessary to estimate the relative shifts between these frames. Affine Scale-Invariant Feature Transform (ASIFT) descriptors were used to compute the relative global displacements in case of single subject, such as writing on a board, a poster, etc. The case of multiple subjects, such as a person in front of a board, has also been tackled in this project. The different subjects were segmented out by suitably thresholding the same depth maps.

The problem of estimation of the de-fenced image is now that of finding out the value of the pixel at which there was fence in the original image, using information obtained from the other images which show the same scene from different angles. This is solved using an optimization-based framework. that models the unoccluded image as a Markov Random Field, and obtaining the maximum a-posteriori estimate (MAP) using Loopy Belief Propagation.

Table of Contents

List of Figures

Chapter 1

Introduction

Chapter 1

Introduction

1.1 Motivation

With the advent of smartphones and tablets, the capturing of images has increased manifold. One of the more pertinent problems faced by photographers is when the subject they want to capture is partially blocked by a fence. Several exhibits, museum showpieces, landmarks, are hidden behind fences and such occlusions. Also, surveillance cameras are installed in locations that view scenes behind fences. For proper surveillance it is imperative that these fences not be present in the images captured. All these reasons provide enough motivation for de-fencing of images.

The method proposed in this project makes use of depth sensors to identify occlusions. Since in recent times low-cost depth sensors are easily available in devices such as Microsoft Kinect, and are expected to be the next step in the upgradation of smartphones/tablets/phablets, the real life application of the work done in this project is relevant and substantial. The primary novelty of this project is the use of depth information to perform fence segmentation, which is much easier than previous methods.

1.2 Objective

The objective of this project is to be able to de-fence images, i.e. remove fences/occlusions from images, thereby revealing the information hidden behind the fences/occlusions and completing the image. Since the proposed algorithm makes use of depth images, a Microsoft Kinect device was used to record both colour and depth images. Experiments were conducted to fulfill this objective with various types of fences, objects in the image and lighting conditions.

1.3 Literature Survey

As shall be discussed later, the most crucial problem in image de-fencing is the detection of the fence in the image. Fence detection is closely related to the image processing problem of the identification of regular and near-regular patterns, which has been tackled in various ways since a long time. Current methods of fence detection range from simpler techniques such as colour segmentation, to highly mathematical methods. Fence detection using colour segmentation is performed by filtering the image to be de-fenced according to the colour of the required fence, by thresholding the pixel intensities accordingly. As can be gathered, this method is extremely naive in its approach, and is riddled with faults for various cases of fences.

Several recent methods employ mathematical methods for fence detection. Leung and Malik [15], Schaffalitzky et al. [25], and Tuytelaars et al. [28] identify visually similar and connected structures under distortion. Forsyth [7] detects and clusters visually similar patterns with lesser regard to their geometric arrangement. Other methods such as those of Hays et al. [11], Liu et al. [18], and Park et al. [23] employ regular and near-regular lattice detectors. The core idea behind these methods is that a fence or an occlusion can be modeled to be a pattern that can be reduced to a repetition of similar structures. The modelling is done using textures or lattices. It can be said that they smartly take advantage of the knowledge of the fence characteristics. These methods are mathematically sound and have been proven to work. However, as shall be seen later, we present a much simpler method for fence detection that is analogous to colour segmentation, but offers results that are comparable to those of the other mentioned methods.

As shall be discussed later, the second step is to find out the shift between the same object in multiple images. Several methods are available to find out global and non-global displacements. However, very few techniques work in the presence of occlusion. One of the most popular methods to find out displacements is using optical flow [1, 26]. This is an optimization algorithm that can accurately find out the shift a particular pixel has gone through between two similar images, assuming the pixel intensity remains the same. Several types of aberrations have been taken care of in its mathematical framework, such as illumination effects and blurring. However, the major drawback of optical flow algorithms is that they cannot handle occlusions well. To be sure, optical flow has been tried during the course of this project, and proven to not work. Of course, the naive way of approaching the problem of calculation of displacements of

pixels between multiple images is to find out the correlation, or related term, between two images while shifting one image row-wise and column-wise and taking those shifts for which the correlation is maximum. This has been tested to work, but with limited resolution, as shall be explored later. Another method that was used is Affine Scale-Invariant Feature Transform (ASIFT) [20, 30]. The beauty of this algorithm is that it is able to detect the pixel locations of the same object in multiple images, irrespective of the parametric distortion the object has undergone. This has proved extremely useful in accurately calculating the pixel shifts, and has been used in this project.

1.4 Scope of Work

Having identified the objective, the scope of work consisted of classifying the different types of images on which de-fencing was to be performed, and capturing the required data sets. Also, since Microsoft Kinect was used it was also imperative to figure out the required softwares to be installed to link the Kinect device to a computer and capture data. Sufficient time was to be allotted to learn to handle the Kinect device, and capture as many types of data as possible.

Once data was captured, it was imperative to design a suitable algorithm to geometrically match the depth data with colour data, since the Kinect colour and depth cameras capture slightly different views of the same real world scene. However, care was to be taken to not distort the original images so much as to render them useless for subsequent mathematical analysis. This meant that only Similarity Transform with non-uniform scaling is allowed, since Affine Transform would include shear transform that would not store the required image features. Sufficient experimentation was required to estimate the best parameters for the transformation.

Experimentation was also required in the algorithms to be used to calculate pixel shifts between the images captured, taking note of the fact that the images concerned contain occlusions. A thorough literature survey was to be conducted to identify and test suitable algorithms. As was mentioned in the literature survey, optical flow algorithms [1, 26] and

ASIFT [20, 30] were to be used for this purpose. Also, the naive approach mentioned was also to be implemented as a backup to simplify matters.

Loopy Belief Propagation [13] was chosen as the optimization algorithm to inpaint the occlusion regions in the images, and the code for that was readily available, since my supervisor had worked on it during his doctorate candidacy. However, the code took quite a long time to run once for one set of images, so there was scope to reduce the running time by optimizing on algebraic and logical operations, and converting them to matrix operations.

Finally, since this project chiefly consists of image processing operations, MATLAB was chosen as the platform to code and run the operations required. Hence, a significant amount of time needed to be spent in writing MATLAB code for all the steps involved in de-fencing of images, including the segmentation of fence from depth map using the Kinect device, matching of depth map to colour map, and finding out pixel shift between multiple images of the same scene.

1.5 Work completed

- Prepared a work flow for capturing images using Microsoft Kinect to de-fence.
- Experimented with external parameters such as number of multiple views, distance between multiple views, types of fences, lighting conditions, subjects, etc.
- Designed a method to approximately scale depth map with colour image, using Similarity Transform with non-uniform scaling; implemented this in MATLAB.
- Captured images of and removed fences for various scenarios:
    - Rectangular fence and a black-board with text written in chalk
    - Rectangular fence and posters with printed text and images
    - Rectangular fence and a multi-textured bedsheet
    - Rectangular fence and a painting

    - Rectangular fence and a person & background
    - Diagonal fence and a poster with printed text and images
    - Diagonal fence and a multi-textured bedsheet
    - Diagonal fence and a logo
    - Diagonal fence and a person & background

- Experimented with pre-processing steps required for fence segmentation, involving performing image processing operations on images captured, in MATLAB
- Experimented with naive algorithm, Optical Flow algorithm and ASIFT to calculate shifts in locations of objects in multiple images.
    - The naive algorithm was designed keeping in mind that pixel intensities should match when the two images are properly aligned over one another. Thus, a variance term was designed to account for the difference in pixel values throughout the image. This algorithm was coded in MATLAB.
    - Optical Flow was being worked on by other students under the supervisor, and their help was sought in operating in on the images captured. It was experimentally verified that optical flow is not capable of handling images with occlusions.
    - A running code for ASIFT [30] was readily available online which was used to compute pixel shifts between multiple images of the same scene. However, an average value was used since the assumption considered for de-fencing is that of global displacement.
- In case of images with multiple layers, more preprocessing was experimented with to segment the front layer from the back layer.
- Wrote all the original code, except the Loopy Belief code, in MATLAB
- Optimized Loopy Belief code: brought down the time taken to de-fence one colour image from around 30 minutes to less than 3 minutes.

1.6 Organization of Thesis

This thesis is partitioned into the following chapters.

Chapter 1 presents an introduction to the problem statement, describes work that has been previously done in this field, details the scope of work in the project, and finally highlights the major achievements made in lieu of completing the objective of the project.

Chapter 2 describes the methodology of the algorithm proposed to reach the objective of the project, as well as the steps involved.

Chapter 3 explains the first step of the algorithm, the detection of fence. It first introduces the Microsoft Kinect and its working, and then describes the procedure used to detect fences in the images to be de-fenced. It also explores the need and method of aligning the depth image with the colour image, and finally presents some failure cases in the detection of fences.

Chapter 4 explains the next step in de-fencing, the computation of Pixel Shifts between the images captured. The concept of Pixel Shift and its usage are introduced, before elucidating on three methods that have been used to calculate it, along with their merits and demerits.

Chapter 5 explains the last step of the algorithm, the inpainting of fence pixels. Loopy Belief Propagation and its basics are introduced, and the mathematical framework of the proposed algorithm is described.

Chapter 6 presents experimental results of the proposed algorithm. It can be observed that the algorithm works for various cases of subjects and fences.

Chapter 7 compares the results of this method with two other recent methods.

Chapter 8 elucidates the precautionary measures taken to ensure the assumptions of the algorithm are not violated.

Chapter 9 details the conclusion of the project, emphasising on the achievements made. It also introduces the future scope of the project.

Bibliography enlists the various references used throughout the duration of the project.

Chapter 2

The Proposed Algorithm

Chapter 2

The Proposed Algorithm

2.1 Methodology

The problem of de-fencing of images has been tackled in various ways, but the important steps involved remain the same. The key tasks to be performed are the detection of the fence, and its removal by inpainting. It is proposed that multiple views of the same scene be used to inpaint the fence regions. Since multiple views would mean that the fence and the scene behind it are captured differently, what is occluded by the fence in one image is revealed in another image. Thus, by capturing multiple images of the same scene with different fence configurations, it is possible to inpaint the fence regions in one image using pixel values from the other images.

Hence, the solution to this problem can be divided into three main tasks:

1. Detection of fence/occlusion in all images.

2. Computation of relative shifts between multiple images.

3. Inpainting of fence region in one image using information from the other images.

As was mentioned earlier, the detection of fences was performed using the depth map that was recorded using the Kinect device along with the colour image. This technique is much simpler than any mathematically robust method, and takes advantage of a technological advancement that is soon to be prevalent.

The computation of shifts between the multiple views was explored using three methods: the naive approach, using optical flow algorithms, and using ASIFT. Ultimately, the naive approach and ASIFT were deemed suitable, since it was experimentally verified that optical flow algorithms are not capable of handling occlusions in images.

inpainting of the fence regions is, at its core, an optimization problem, and in this project, Loopy Belief Propagation was used to perform said optimization, since it is a fast-running technique that is extensively used in computer vision.

The primary novelty of the proposed algorithm is the use of depth maps for the de-fencing of images. This is a relatively unexplored area in the field of research and has a lot of potential for future advancements. The algorithm described above and the experimental results brought using the algorithm have been compiled into a research paper and submitted to ACM Multimedia 2014, the 22nd ACM International Conference on Multimedia.

In the subsequent sections, the proposed method for de-fencing of images is explored step-by-step, by elucidating each of the above steps in detail.

2.2 Workflow

- Capture multiple images of the same scene using a Kinect device, by slightly translating the camera in a plane normal to the direction of vision.
    - Care is to be taken to take enough number of images so as to reveal all the information required. Generally, 4 images should be adequate, so long as there is enough pixel shift between the images. This is explained in Chapter 7..
- Detect the fences in the images captured, as explained in the next chapter.
- Among the multiple images captured, choose one image as the "Reference Image" to be de-fenced.
- Barring the fence region, calculate the pixel shifts between the scene in the Reference Image and the other images, as explained in Chapter 4.
    - In case of multiple layers within the scene, pixel shift needs to be calculated for each layer.
- Using the calculated shifts, inpaint the fence regions in the Reference Image using Loopy Belief algorithm, as explained in Chapter 5.

Chapter 3

Detection of Fences

Chapter 3

Detection of Fence

3.1 Microsoft Kinect

The Microsoft Kinect is an image capturing device that records images in two domains: colour (RGB) and depth (D), hence the combined data captured is termed RGB-D data. A Kinect device has an infrared emitter, and two cameras on board, one for recording colour data, and the other an infrared detector for capturing depth data.

The colour camera is a standard visible spectrum recording device, much like any digital camera or webcam, with an image resolution of 640x480 pixels. The infrared camera consists of infrared detectors that capture the infrared rays that are reflected off objects in the scene that is being captured. Since there is an infrared emitter that is present on the Kinect device, it is assumed that the infrared detected by the camera is emitted by it.

The infrared emitter throws an array of dots of infrared light at the scene to be recorded. These infrared rays get reflected off objects in the scene, and are captured by the infrared camera on the Kinect device. By measuring the time lag between the emission of a ray of infrared light and its capture, the distance between the object in the scene the infrared ray is reflected off of and the Kinect device is gauged. This distance is termed "depth," since it denotes the depth of an object from the Kinect device.

The measurement of depth is a relatively new area in technology since the technical feasibility of it had always been limited, until the advent of Microsoft Kinect and related devices. Hence, research work in the use of depth data is upcoming and relevant in the modern age. Microsoft Kinect offers a handy, cost-effective way of computing depth, and for this reason it has been welcomed in the research community. Fields such as gesture recognition benefit significantly from depth-sensing devices, and so the Kinect and depth-sensing is being applied to research areas that have been tackled before too.

This project is one such attempt at applying depth-sensing to the problem of de-fencing of images. As shall be seen in the next section, this application significantly simplifies the first step in de-fencing, i.e. the detection of fences.

It is important to mention that the colour and depth image captured are not perfectly aligned with each other, i.e. the pixel location of a real world object in the colour and depth images are not exactly the same. This is because the structures and locations of the two cameras are different, so they record slightly different views of the same scene. The good news is that this can be modeled as a geometric transformation, and the depth image can be made to perfectly coincide with the colour image.

3.2 Capture of Colour and Depth Images

As explained in Section 2.2, multiple images of the same scene need to be captured. This means that the camera, in this case a Kinect device, needs to be positioned at different points to capture multiple views of the same scene. However, care is to be taken not to shift the camera's position too much from that used to capture the Reference Image, since otherwise the result of inpainting would be blurry. This shall be explained later in Section 4.2.

3.3 Fence Detection

As discussed earlier, the primary novelty of this project is the use of depth maps. Since depth maps basically display the distance of an object from the camera, once the images have been captured, the fence can be segmented out by simply thresholding the depth map suitably.

As an example, displayed in Figure 3.1 are the colour and depth images captured by the Kinect device, and the fence that has been segmented out from the depth image after aligning it with the colour image.

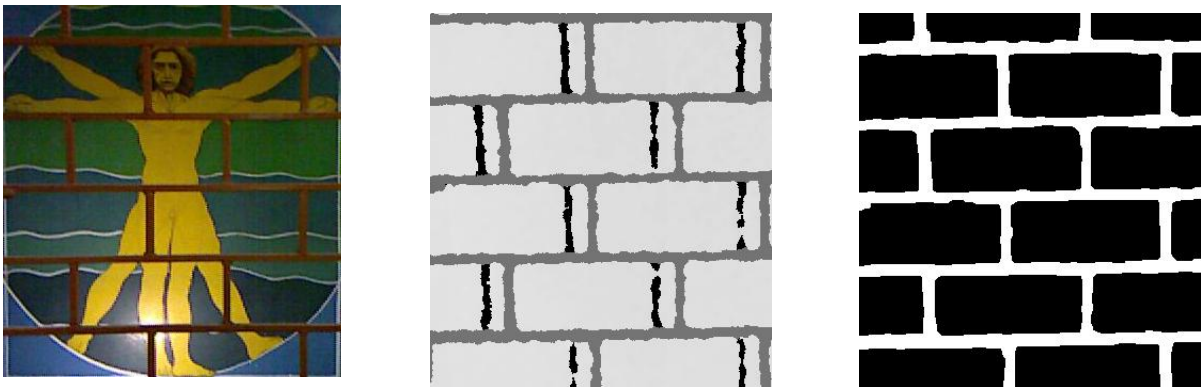
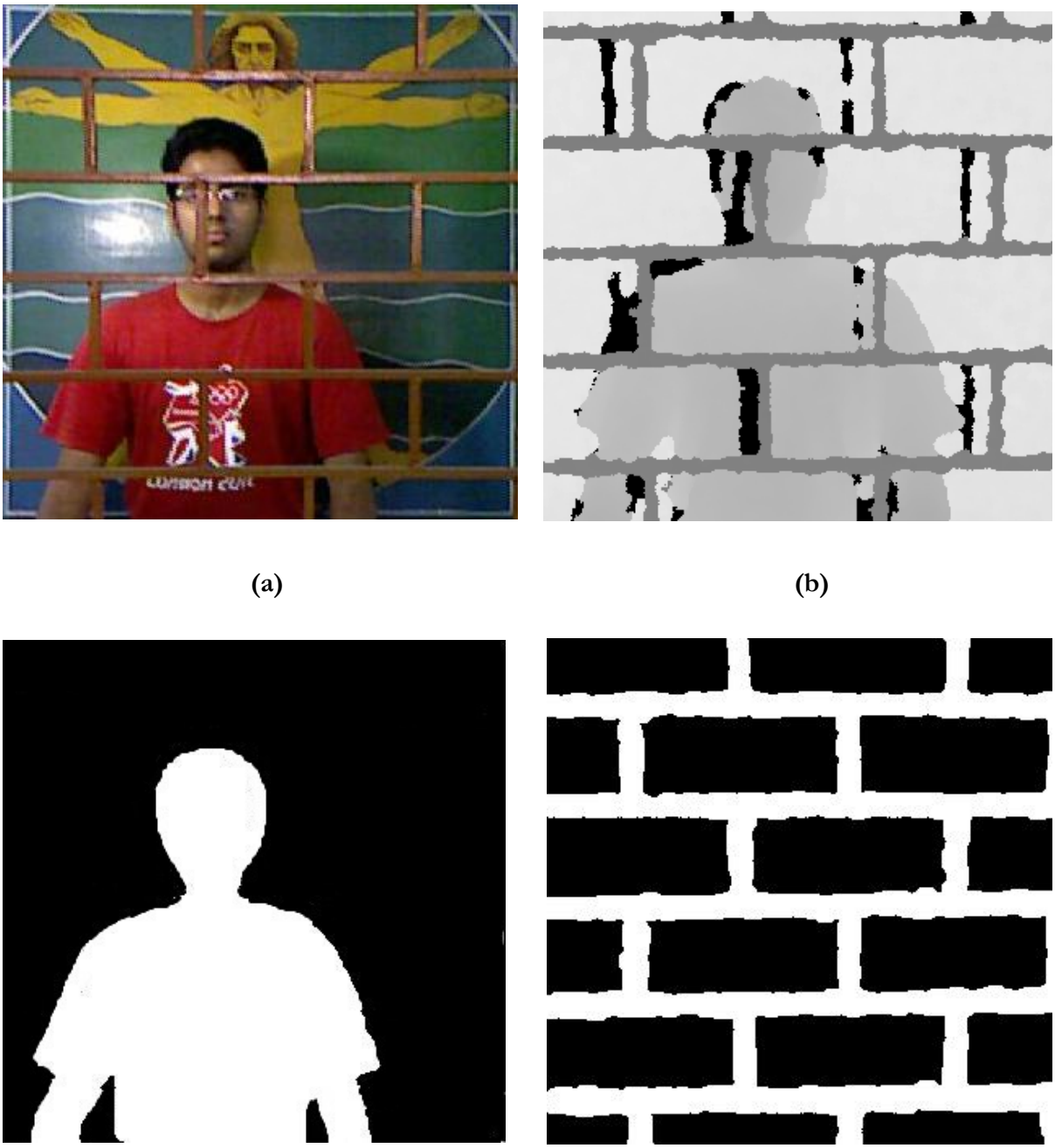


Figure 3.1: (a) Colour image. (b) Depth image. (c) Fence segmented out from depth image.

As can be clearly seen in the depth map, the light grey is the painting's distance from the Kinect device, while the dark grey is the fence's distance. It can also be seen that the fence in the depth map does not align with the fence in the colour image. As discussed in the previous section, the reason for this is that the location and structure of the depth camera is slightly different from that of the colour camera. Since the alignment of the depth map with the colour map is crucial for any future use of the depth map, a method to achieve it is was designed, and is discussed in the next section.

The third image is the result of thresholding the depth map for the fence's intensity values, and then aligning it with that of the colour image (this shall be discussed in the next section). In addition, some preprocessing operations were performed to smooth out the fence and give it a less haphazard shape than the one seen in the depth map. The image processing operation of opening the thresholded image serves to remove the noise in the image, and closing serves to fill pixels within the fence that might have been missed in the depth map. Finally the fence pixels are then dilated, so as to cover an area larger than the original fence. The reason for this shall be discussed in Section 7.1.

On a sidenote, the black regions in the depth map represent where the fence's shadow falls on the painting from the perspective of the depth camera. Since this region does not reflect any infrared light back to the depth camera, it is not visible to it. A default value of 0 is assigned to those pixels.



(a) **(b)**

(c) **(d)**

Figure 3.2: (a) Colour image. (b) Depth map. (c) Foreground (person) segmented out from the depth map. (d) Fence segmented out from the depth map.

3.4 Alignment of depth map with colour image

(a) (b)

Figure 3.3: (a) Fence Image segmented out of depth map overlapped on colour image. (b) After aligning fence image with fence in colour image.

The alignment of the fence image with the colour image is a critical step in moving forward with de-fencing of images using the proposed algorithm. Figure 3.3 (a) shows the fence segmented out of the depth map, overlapped with the colour image. It can be clearly seen that the fence in the depth map does not coincide with the fence in the colour image. Thus it can be concluded that the depth image itself does not coincide with the colour image, in terms of pixel locations. In order to align the depth image with the colour image, some transformation is required that preserves the important characteristics of the depth map, yet aligns it with the colour image.

A simple way of performing this is to use a Similarity Transform on the depth map, along with non-uniform scaling. Since Similarity Transform does not perform Shear Transformation or any composition of Perspective Projections, it does not significantly alter the characteristics of the depth image. However, Similarity Transform does not perform non-uniform scaling, since it preserves lengths. Various experiments were performed by using Similarity Transform, and it

was concluded that in order to achieve alignment, non-uniform scaling is necessary, albeit of a small magnitude.

Thus, a methodology for transforming the depth image to align with the colour image was made. The steps involve scaling in different dimensions one after another to achieve maximum possible alignment in the fastest way possible. As the depth map is scaled, the variance of the pixels in the colour image that coincided with the fence pixels found from the depth image is calculated. It is assumed that the fence pixels in the colour image have more or less the same pixel intensities. Hence, the depth map coincides maximally with the colour image when for every scaling performed the value of this variance is minimum.

The different scalings performed are (1) Scaling in proportion, i.e. uniform scaling, (2) Scaling horizontally leftwards keeping the right edge constant, (3) Scaling vertically upwards keeping the bottom edge constant, (4) Scaling horizontally rightwards keeping the left edge constant, (5) Scaling vertically downwards keeping the top edge constant. In some cases, (4) and (5) are prove to be redundant, but that is to the discretion of the experimenter. Figure 3.4 shows the operation of these Scalings on the Reference Image, and Figure 3.3 (b) displays the result of this alignment.

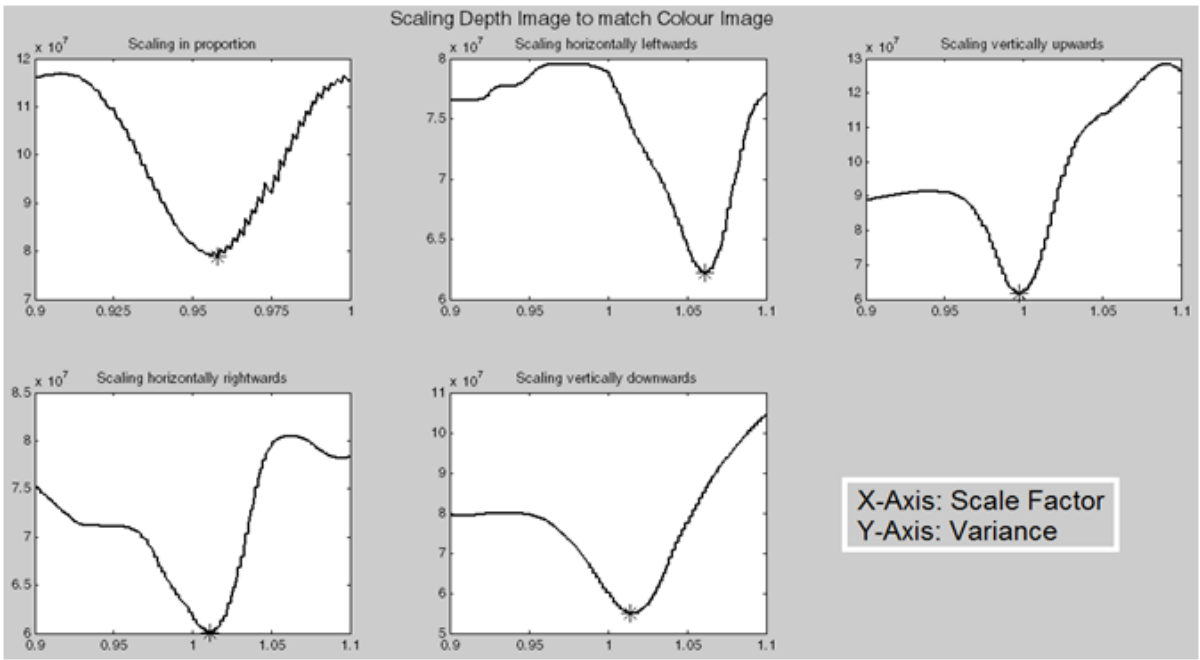


Figure 3.4: Variance versus Scaling Factor for the different scalings

3.5 Failure in Fence Detection

Since fence detection is the most important step in de-fencing of images, it is noteworthy to note when the procedure described hitherto fails in detecting occlusions. One primary aspect that is to be considered is that since we are dealing with a device that detects infrared rays, the procedure is susceptible to failure when the infrared detection suffers. This could be due to several reasons, but the basis lies in radiometry.

Failure to detect infrared rays could be due to fault in the reflection of infrared rays from an object, or due to over-exposure to infrared rays, possibly due to leakage from the surroundings. These form the two chief two reasons for why the fence in the image cannot be detected, and are elucidated below.

3.5.1 Minimal Reflection from fence

Since the fence is a physical object, it is possible that the radiometric parameters and/or the physical attributes of the fence are not congenial with reflection of infrared light. For example, Figure 3.5 shows a metal fence that is too thin to be able to reflect enough infrared light so as to be able to get detected by the Kinect's depth camera.

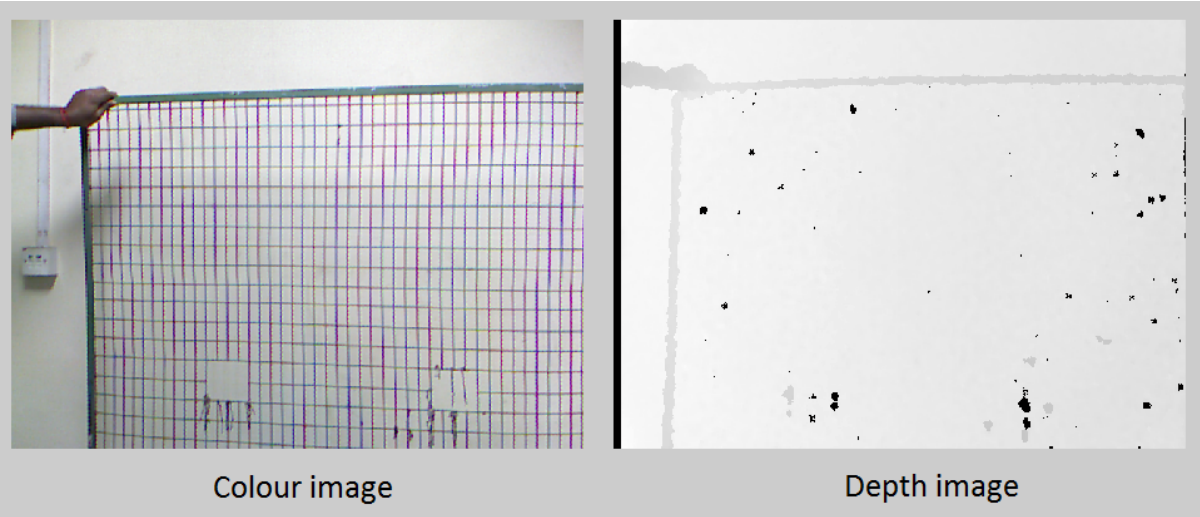


Figure 3.5: The colour and depth images of a fence that is too thin to be detected in the depth map.

As can be clearly seen in the Figure 3.5the metal fence used was too thin to be seen by the Kinect's depth camera. The edge of the fence is visible since it has enough thickness, but the grill structure is not at all visible.

3.5.2 Over-exposure to infrared light

The major drawback of infrared detectors like the Kinect is that they are meant mainly for indoor use. infrared detectors that are open to outdoor environment are typically used for obstacle detection, in which case they need not possess accuracy like the Kinect requires to. The chief reason for this is that in an outdoor area, there is too much of interference for the Kinect's infrared camera from the infrared rays emitted by the sun. As an example, observe the depth image that was captured in an outdoor environment, in Figure 3.6

(a) **(b)**

Figure 3.6: The (a) colour, and (b) depth images, captured in an outdoor environment.

As can be clearly seen, the major contributor to the depth map is infrared light from the sun. This makes it extremely difficult to be able to use the Kinect in an outdoor environment, and thus limits the scope of experimentation, at least using a Kinect device.

Chapter 4

Computation of Shifts between Images

Chapter 4

Computation of Shifts between Images

4.1 "Pixel Shifts"

As was mentioned in the second chapter, the proposed algorithm makes use of pixel information from multiple images to inpaint the fence region in the Reference Image, which is one image that is chosen to be de-fenced among the multiple images captured. Thus, in this respect, in order to inpaint the fence pixels in the Reference Image, information about the intensity values of those pixels is required. This information is borrowed from the other images of the same scene that are captured. This is the reason why multiple images are recorded. Only if there is at least one image in which each fence pixel that needs to be inpainted is unoccluded can this method work.

However, simply recording multiple images is not sufficient to be able to inpaint properly. Consider one fence pixel in the Reference Image whose intensity value needs to be replaced with that corresponding to the scene it is occluding. It is understood that among the other images recorded, there shall be images where this pixel's information is unoccluded. Let us call such an image a "Test Image." However, it is not known which pixel location in the Test Image corresponds to the pixel in the Reference Image being considered. In all likelihood, it is not in the same location, since while recording the images the Kinect device was moved, as discussed in Section 3.2. Thus, there is a need to calculate the row-wise and column-wise displacement of the pixels between the Reference Image and the Test Image. This displacement is referred to as the "Pixel Shift."

A "layer" in an image refers to the images of a collection of objects that lie in the same plane in the real world normal to the direction of vision of the camera. For example, in Figure 3.2, the fence represents one layer, the person represents another layer, the Foreground, and the painting represents the third layer, the Background. Hence, with the translation of the camera,

it is assumed that all the Foreground pixels are shifted by the same Row Shift and Column Shift. The same goes for all the Background pixels.

The Pixel Shift for the foreground and background are not the same, though. To understand this, consider watching through the window of a moving train. With some movement of the train, an object, say a tree, that is close to the train appears to move by a smaller amount than an object, say the moon, that is farther away. In this case, the tree can be considered to constitute one layer, and the moon another. In the same way, different layers in the images captured shift with different values for Row Shift and Column Shift.

In order to calculate pixel shifts, three methods have been tried:

1. The naive algorithm: This is the first algorithm that would come to mind when trying to achieve the objective of calculating pixel shifts.

2. Optical Flow algorithm: This is a prevalent technique to calculate shifts between images, especially in motion estimation. However, as shall be seen later, it cannot handle occlusions.

3. ASIFT: This is a robust algorithm that can be used to accurately calculate shifts for every pixel in the Reference Image. It is readily available as an online tool on the website of Image Processing On-Line (IPOL) [30].

4.2 Assumption of Global Motion

An important point to note is that it is assumed that within a layer, the values of row shift and column shift are the same for all pixels. This means that it is possible that the translation of the camera, i.e. the Kinect device, could be such that Pixel Shifts are different for every pixel, even within a layer. However, we assume that the camera has moved such that the value for Pixel Shift is the same for all pixels within a layer.

This is a valid assumption since the distance between the camera and the layers are significantly large. This allows for ignoring the tiny differences in Pixel Shift within a layer, since the marginal error incurred is readily acceptable compared to the theoretical and

computational complexity and analysis that would otherwise be required to take further steps in de-fencing the Reference Image.

Moreover, in order to give this assumption it's share of validity, during recording care is taken to only translate the camera in a plane perpendicular to its direction of vision. This ensures that no unwanted projective transformations occur between the images, which could otherwise invalidate the assumption.

In addition to the above steps, care is taken not to translate the camera too much, leading to higher Pixel Shifts. This is because camera translation inherently adds perspective distortion to the scene, which does not fit into the assumption of simple row and column shifts, which shall lead to bad results. Thus, the camera is moved within a close distance of the camera position used to capture the Reference Image, but enough so as to reveal the information behind the fence. Section 7.2 describes the situation when the translation is not enough.

4.3 The Naive Algorithm

The objective is to find a function that can describe the non-alignment of the Reference Image and one of the other images, the Test Image. Using a trial-and-error based method, the Test Image is shifted row-wise and column-wise. For any value of Row Shift and Column Shift, let us call the resulting image the Shifted Test Image. Then, the variance of the image formed by the difference of the Reference and Shifted Test Images is computed for all those pixels that do not belong to a fence in either image. This value is divided by the total number of non-fence pixels raised to a convenient power in order to normalize it. The power also allows for scaling of the graph formed, which helps in finding out the minimum both mathematically and visually. Let us call this value the Error Variance. Thus, the Error Variance is minimum when the two images are aligned.

Error Variance = variance[non-fence(Reference_Image - shift(Test_Image))]/(number of non-fence pixels)^(a convenient power).

The values of Row Shift and Column Shift for which the Error Variance is minimum describe the relative displacement, i.e. the Pixel Shift between the Reference Image and the Test Image.

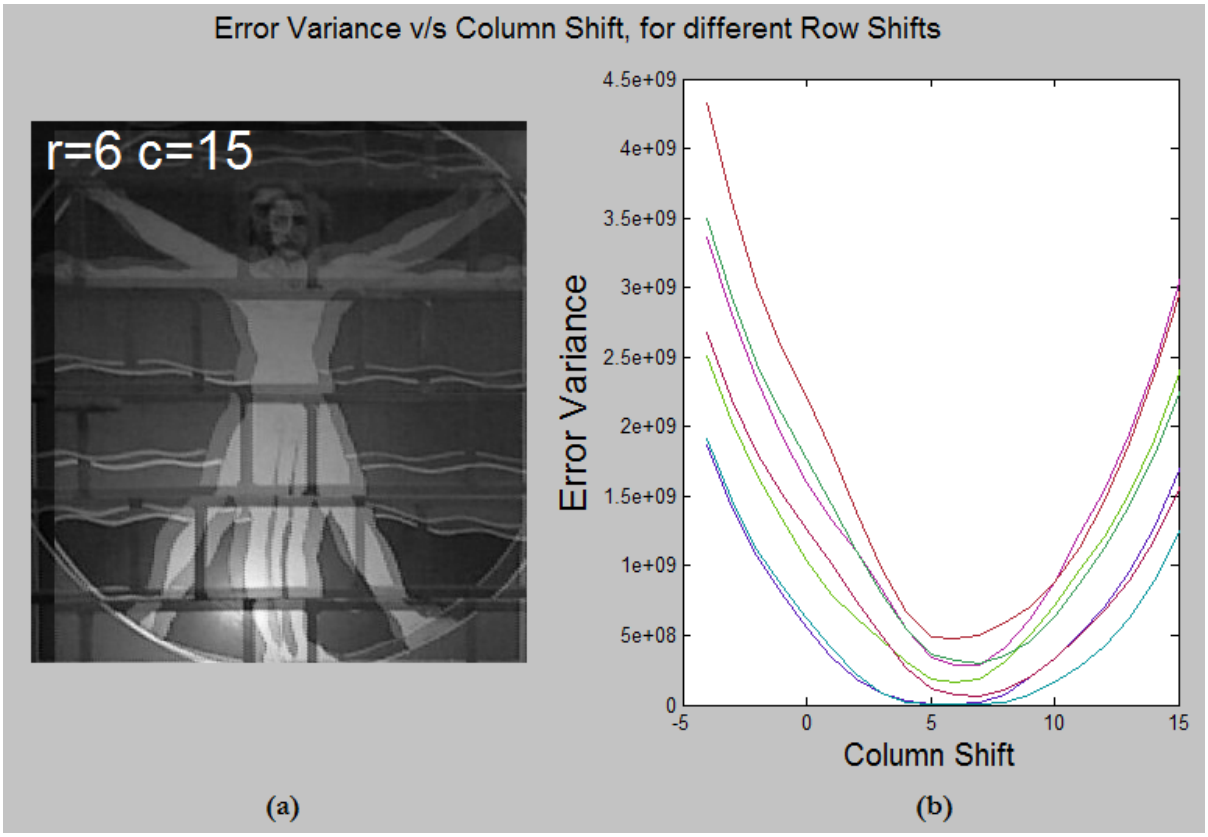


Figure 4.1: (a) Overlap of Reference Image and Shifted Test Image while testing. (b) Variation of Error Variance with row shift and column shift. (c) Overlap of Reference Image and Test Image shifted by values calculated by naive algorithm.

Figure 4.1 displays the working of the naive algorithm. Figure 4.1 (a) shows the overlap of the Reference Image and the Shifted Test Image for a Row Shift of 6 pixels and a Column Shift of 15 pixels. In this way, using trial and error, the Test Image was shifted to check alignment with the Reference Image, in this case over a range from -5 pixels to 5 pixels of Row Shift, and -5 pixels to 15 pixels of Column Shift. Figure 4.1 (b) displays a graph drawn of the Error Variance calculated for particular values of Row Shift and Column Shift, versus the Column Shift, and the different colours denote different values of Row Shifts. So, one trough-like structure of one colour represents the variation of Error Variance with Column Shift, for one value of Row Shift.

The minimum value of Error Variance was found for 0 pixels of Row Shift and 5 pixels of Column Shift. This is in agreement with how the images were recorded, since the Test Image was captured after shifting the Kinect device horizontally after capturing the Reference Image. To further illustrate the correctness of the algorithm, Figure 4.1 (c) displays the overlap of the Reference Image and the Shifted Test Image with values of pixel shift as those corresponding to minimum Error Variance. As can be clearly seen, there is hardly any disparity in the coincidence of the two images. It can also be observed that the fences in the two images do not overlap, as explained in the previous section, but this is not a problem since the objective is to eliminate the fences. But indeed, it is interesting to note that where the fence is present in the Reference Image, the pixel information is revealed in the Test Image. This is precisely the observation that the proposed algorithm takes advantage of. When it comes to the vertical sections of the fence, pixel information can be directly borrowed from the Test Image. However, the horizontals sections are occluded in both images, and so more recordings with fences shifted vertically are required to inpaint in those regions.

Although this provides the simplest way of computing the Pixel Shifts, it is highly time-consuming since it is based on trial-and-error. Moreover, the values for Row Shift and Column Shift obtained are integral. Inter-pixel lengths cannot be handled by this algorithm. So there are high chances of fractional errors in the true Pixel Shifts. This may not be too much of a problem if the pixel size is small enough, as it generally is, but it is worth mentioning.

4.4 Optical Flow Algorithm

Optical Flow [1, 26] is one of the most popular methods in literature to find out Pixel Shifts. It is primarily used for motion detection, when there is relative motion between the observer and a scene. It is an optimization algorithm that accurately finds out the shift a particular object has gone through between two similar images, in terms of pixel location. Several types of aberrations have been taken care of in its mathematical framework, such as illumination effects and blurring. These effects are handled since the algorithm looks for features such as Eigenvalues, that do not depend on the aberrations mentioned.

However, the major drawback of optical flow algorithms is that they cannot handle occlusions well. The features extracted are not localized enough to account for occlusions, and are highly dependent on neighbouring pixel values. Thus, in the presence of occlusions such as fences, optical flow algorithms fail in estimating Pixel Shifts. To be sure, optical flow has been tried during the course of this project, and proven to not work.

4.5 Affine Scale-Invariant Feature Transform (ASIFT)

ASIFT [14, 20] is an acclaimed algorithm that is used to find out correspondence points between two images. The result of the algorithm is that pixel locations of identical objects in both the images are identified. ASIFT is an improved version of the SIFT algorithm. Whereas SIFT employs descriptors or features extracted from the image to account for rotational and translational effects between the two images, ASIFT crosses the bridge to account for affine transformation between the two images. Thus, by identifying the same feature points in two images, the displacement between the pixels of the two images can be measured.

According to Wolfram Mathworld, an affine transformation is any transformation that preserves collinearity (i.e., all points lying on a line initially still lie on a line after transformation) and ratios of distances (e.g., the midpoint of a line segment remains the midpoint after transformation). Affine Transformation consists of six degrees of freedom, and

so allows for six types of change in an image, while preserving collinearity and ratio of distances. The transformations allowed are:

1. Translation in two dimensions

2. Rotation

3. Uniform Scaling

4. Non-uniform Scaling

5. Shear Transformation

The above transformations lend the six degrees of freedom to Affine Transform. The only transformation that is not allowed is Perspective Projection, which, if included, would make it Projective Transform. The problem of finding out correspondence points between two images reduces itself to the problem of the computation of affine invariant image local features.

By simulating zoom out and normalizing translation and rotation, SIFT is invariant to only four out of the six parameters of an affine transform. ASIFT not only covers the four parameters taken care of by SIFT using SIFT itself, it also simulates all image views obtainable by varying the two camera axis orientation parameters, namely, the latitude and the longitude angles. The resulting method has been mathematically proved to be fully affine invariant.

Hence, this forms enough mathematical background check to use ASIFT for the purpose of computation of Pixel Shift between two images. A running code for ASIFT is readily available on the website of Image Processing On-Line (IPOL), provided by the creators of ASIFT, Guoshen Yu, and Jean-Michel Morel, at http://dx.doi.org/10.5201/ipol.2011.my-asift. All that needs to be done is to upload the two images between which Pixel Shift needs to be found out, and the algorithm shall output a matrix of two sets of row and column values, corresponding to the pixel locations of identical points on the two images.

The values gotten are highly accurate, and contrary to the result of the naive algorithm, may refer to sub-pixel motion. As an example, Figure 4.2 displays a few of the points of correspondence obtained by using ASIFT on the two images shown.

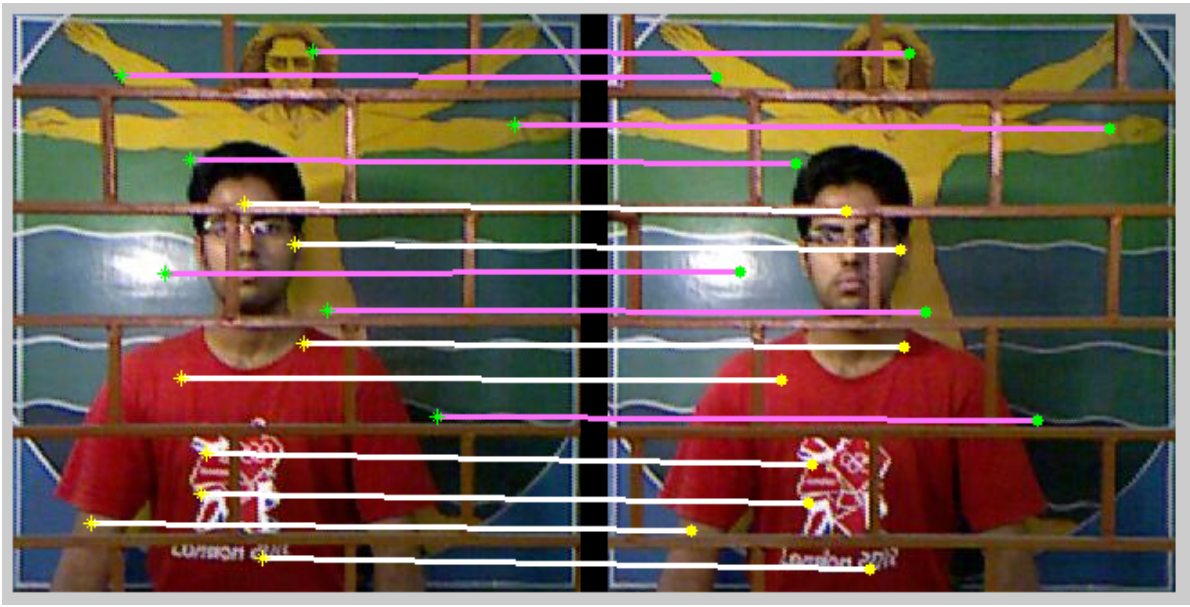


Figure 4.2: Matching identical locations between two images using ASIFT

It can be observed that Figure 4.2 contains images with two layers apart from the fence: the Foreground, i.e. the person, and the Background, i.e. the painting. It is natural to assume that different values for Pixel Shifts must be obtained for the two layers. Hence, the correspondences are identified with different colours: white for the Foreground and pink for the Background.

The output of ASIFT gave accurate values for pixel shifts that were correct to 4 decimal places. Many different values were obtained for different pixels even within the same layer. However, since the assumption is one of global displacement for each layer, the average value of the Pixel Shift of pixels from the same layer was used as the pixel Shift for that layer.

The advantage of this method is that it is purely mathematical, and so can result in more accurate values for the displacements, as compared to the integral values provided by the previous method.

Chapter 5

Inpainting of Fence Regions

Chapter 5

Inpainting of Fence Regions

In order to inpaint in the pixels in the Reference Image that are occluded by the fence in it, now that the Pixel Shifts with the other Test Images are known, it is necessary to smartly combine the pixel information from the Test Images. This is done with the help of Loopy Belief Propagation. Belief Propagation is a technique that is used to infer and pass on information in graphical models. This method is used for the purposes of this project by assuming the images to be Markov Random Fields.

5.1 Markov Random Fields

A Markov Random Field is an undirected graph of random variables that exhibit the Markov property, i.e. the value of a node only depends on the values of its immediate neighbours. In most image processing applications, the underlying image that needs to be discovered is modelled as a Markov Random Field. For example, in the field of image restoration, given a blurred image the underlying sharp image is modelled as an MRF. In the case of this project, given the fenced Reference Image, the underlying de-fenced image is modelled as an MRF.

The usage of Markov Random Fields in image processing has been prevalent since long, since they are known to give good, flexible, stochastic models. A guiding insight underlying most of the work on Markov Random Fields in image processing is that the information contained in the local, physical structure of an image is sufficient to obtain a good, global representation of it. This notion is captured by means of a local, conditional probability distribution, which describes the MRF.

As described in [24], once the local, conditional probability distribution of the MRF is specified, there are five remaining steps involved. First the joint distribution of the MRF is obtained through prior information. In this way, the image is represented in one global, joint

probability distribution. Next, the process by which the observed image is obtained from the image modelled as an MRF is captured in a degradation probability distribution. In the image restoration problem mentioned above, this degradation is blurring. In the problem statement of this project, the degradation model, given in the next section, is the presence of occlusions in the image, as well as the row and column shifts in case of Test Images, modelled as a Warp Matrix.

Next, Bayes' Theorem is invoked to obtain the posterior probability distribution of the observed images, in this case the Reference and Test images. This posterior distribution gives the probability that an image could have been degraded to obtain the observed image. Once the posterior probability is obtained, a cost function is associated with every possible configuration in the posterior. This cost function describes how far an estimated image is from the true underlying image. Thus, the final step is to minimize this cost function with respect to the parameters of the image, typically the intensity values. In the next section, this procedure is described for the current objective.

5.2 De-fencing Degradation Model

It has been mentioned that the de-fenced image is modelled as a Markov Random Field, and a degradation is applied on it to obtain the observed Reference and Tests Images. The point of choosing a reference image is that it is considered to be the fenced version the de-fenced image. That is to say that removal of fence from the Reference Image should theoretically directly lead to the de-fenced image. The degradation model used is:

$$\mathbf{y_m = O_m.W_m.x + n_m}$$

where $\mathbf{m}$ is a variable used to denote the image number, given there are multiple Test Images and a Reference Image; $\mathbf{y_m}$ represents the observed image, i.e. the Reference and Test Images; $\mathbf{x}$ is the de-fenced image; the operator $\mathbf{O_m}$ crops out the un-occluded pixels from the $\mathbf{m}^{th}$ image, or applies the appropriate fences/occlusions on $\mathbf{x}$ corresponding to the $\mathbf{m}^{th}$ image; $\mathbf{W_m}$

is the warp matrix that describes the Pixel Shift between the observed image and the de-fenced image; and $\mathbf{n_m}$ is noise, assumed as Gaussian.

Thus, the problem statement now becomes that of finding out $\mathbf{x}$, given the different $\mathbf{y_m}$'s, $\mathbf{O_m}$'s and $\mathbf{W_m}$'s.

It is important to note that the Warp Matrix $\mathbf{W_m}$ denotes the Pixel Shift required to translate the de-fenced image to coincide with the $\mathbf{m^{th}}$ observed image. This is nil in the case of the Reference Image, and equal to the Pixel Shifts calculated in the case of the respective Test Images. Thus it can be seen that the assumption of global motion is mathematically used here, where the Warp Matrix carries the Row and Column Shifts that are the same values for each layer, denoting only translation for every layer, and no other transformation. Also, the operator Om is the matrix that adds fences to the de-fenced image to convert it to the observed image $\mathbf{y_m}$. Thus, this is the fence image of $\mathbf{y_m}$ that has been extracted from its depth map. Indeed, when trying to find out $\mathbf{x}$ from the different $\mathbf{y_m}$'s, the complement of the fence images are used as $\mathbf{O_m}$, and the negative of the Pixel Shifts are used as $\mathbf{W_m}$.

The de-fenced image $\mathbf{x}$ is found out iteratively as the maximum a-posteriori estimate calculated by minimizing a cost function. As described in the previous section, the cost function needs to describe the disparity between the de-fenced image and the estimated de-fenced image. Thus, the iterative minimization of the cost function with respect to the estimate of the de-fenced image, for the different observations shall lead the estimation of the de-fenced image closer and closer to the true de-fenced image that is modelled as an MRF. Going about this in another way, the cost function could instead describe the disparity between an observed image and the image obtained by applying the degradation model on an estimation of the de-fenced image. The iterative minimization of this cost function term with respect to the estimate of the de-fenced image should also lead to the same true de-fenced image. In optimization terminology, this term is called the "data term."

$$\textbf{Data term} = ||\mathbf{y_m} - \mathbf{O_m}.\mathbf{W_m}.\mathbf{x}||^2$$

The data term represents the objective to be achieved. However, a good cost function cannot only contain a data term which tries to blindly optimize its value. It must also contain a term to

account for prior knowledge about the estimation. This term that accounts for prior knowledge is called the "smoothness term." The smoothness term helps in leading the estimate towards values that satisfy the prior knowledge, and hence helps in eliminating unwanted estimates. Thus, the total cost function is a sum of a data term and a smoothness term.

$$\mathbf{J} = \mathbf{J}_{\mathbf{data}} + \mathbf{J}_{\mathbf{smoothness}}$$

In the case of this problem, the prior knowledge is the fact that the de-fenced image is modelled as a Markov Random Field. As described in the earlier section, a Markov Random Field model assumes that the image is locally smooth, except for relatively few intensity gradient discontinuities at region boundaries or edges. Hence the smoothness term must account for this local smoothness, and attribute higher values in the cost function for non-smoothness within a pixel neighbourhood. A term that possesses this property is the Clique Potential function $\mathbf{V_c(x)}$, which is given by:

$$\mathbf{V_c(x)} = |\mathbf{x_{i,j}} - \mathbf{x_{i,j+1}}| + |\mathbf{x_{i,j}} - \mathbf{x_{i,j-1}}| + |\mathbf{x_{i,j}} - \mathbf{x_{i-1,j}}| + |\mathbf{x_{i,j}} - \mathbf{x_{i+1,j}}|$$

where $\mathbf{x_{i,j}}$ represents the pixel in the i^{th} row and j^{th} column of $\mathbf{x}$, and $\mathbf{x_{i,j+1}}$, $\mathbf{x_{i,j-1}}$, $\mathbf{x_{i-1,j}}$ and $\mathbf{x_{i+1,j}}$ denote its corresponding neighbour pixels.

Hence, combining the above concepts, the cost function that is to be minimized with respect to the estimate of the de-fenced image is:

$$\mathbf{J(x)} = ||\mathbf{y_m} - \mathbf{O_m}.\mathbf{W_m}.\mathbf{x}||^2 + \beta \sum_{c \in C} \mathbf{V_c(x)}$$

where β is the regularization parameter. It is used to scale the smoothness term properly so as not to allow unequal weightage to either term. In the experiments conducted, the value of β was chosen as 5e-4.

Hence, the true de-fenced image $\hat{x}$ shall be that value of $\mathbf{x}$ that minimizes the above $\mathbf{J(x)}$. This value of $\mathbf{x}$ is found out by optimizing $\mathbf{J(x)}$ using Loopy Belief Propagation.

$$\hat{x} = \mathbf{arg\ min_x}\ (\mathbf{J(x)})$$

5.3 Loopy Belief Propagation

Belief Propagation is a popular technique used to solve graphical inference problems. It is a dynamic programming approach to answering conditional probability queries in graphical models. Hence it makes good sense to use Belief Propagation to solve the optimization problem given in the previous section.

The appeal of Belief Propagation lies in its optimality in solving tree-structured graphical models, for example, Bayesian Networks. When the graph is a tree, i.e. when there are no loops in the graph, Belief Propagation provides the exact solution. However, when the graph is loopy, only an approximate solution can be obtained. It is known that Markov Random Fields do contain loops in the graph that is the array of pixel intensities. Thus, applying Loopy Belief Propagation [13] shall result in an approximate solution, which keeps getting better with the number of iterations it is passed through.

Belief Propagation works through the passing of message across neighbouring nodes in the graph, or in this case, neighbouring pixels in the image. The "belief" that is referred to here is that conditional probability of the intensity of a pixel. This intensity value is decided upon by means of passing of messages between its neighbouring pixels. The messages contain information about beliefs of pixel intensities, based on prior information, conditional probabilities and actual pixel intensity values. These beliefs keep getting updated as more information keeps getting passed as messages. These messages are iteratively passed between all the pixels in an image, until a stable belief state is reached.

In this way, Loopy Belief Propagation is able to inpaint in the fence pixels, by using information passed on from neighbouring pixels. The code for Loopy Belief Propagation was readily available, since the supervisor of this project had worked on it during his doctorate. It is worthy to be mentioned that efforts were put to optimize the available code. Consequently, whereas it used to take more than 30 minutes to inpaint a Reference Image using Test Images, after optimization of the code it took less than 3 minutes for the same job.

The results of using the whole process that has been described hitherto are shown in the next Chapter.

Chapter 6

Experimental Results

Chapter 6

Experimental Results

In the following figures, the image on the left is the reference image out of the four images that have been captured, and the one on the right is the image formed after de-fencing the reference image on its left.

6.1 Rectangular fence with single object: Board

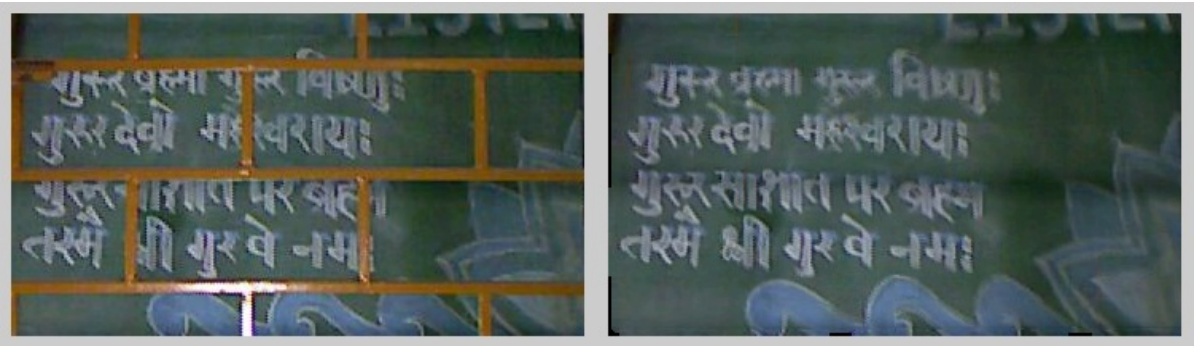


Figure 6.1: De-fencing of image with rectangular fence and a board.

6.2 Rectangular fence with single object: Bedsheet

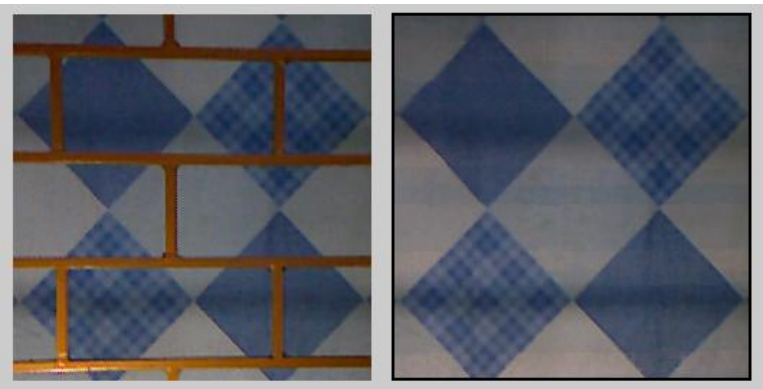


Figure 6.2: De-fencing of image with rectangular fence and a bedsheet.

6.3 Rectangular fence with single object: Poster

Figure 6.3: De-fencing of image with rectangular fence and a poster.

6.4 Rectangular fence with single object: Painting

Figure 6.4: De-fencing of image with rectangular fence and a painting.

The above images were for the case of rectangular fence and single object. Since in an image there is only one object, there is only global displacement to take care of. In the following experiments, images with two layers, a foreground and a background, are considered. Also, a case of diagonal fence is also displayed.

6.5 Rectangular fence with two objects

Figure 6.5: De-fencing of image with rectangular fence and two objects.

6.6 Diagonal fence with two objects

Figure 6.6: De-fencing of image with diagonal fence and two objects.

Chapter 7

Comparison with Other Methods

Chapter 7

Comparison with Other Methods

Any critical analysis of a proposal cannot be complete without a comparison of the results of the proposed methodology with those of existing techniques. In lieu of this, comparisons have been drawn between the results of the algorithm proposed in this project with those of two other recent methods that carry out the same function.

7.1 Total Variation Inpainting using Split Bregman [8]

This method is referenced in [8], and is authored by Pascal Getreuer. This paper explores the problem of inpainting as a minimization problem that uses total variation as a part of the cost function to optimize. Although this does give better results than its predecessors in the same field, it does not work correctly for several cases of real-life fences. The Image Processing On-Line website that hosts the demo of this method contains examples where de-noising has been implemented in various images. But the crucial fact to note is that Split Bregman uses neighbourhood data in order to in-paint, and if the neighbourhood data is not present, it shall not be able to fill the occlusions properly.

The fences considered in our experiments are significantly thicker, or more well-spread, than the test fences used in the experiments conducted in [8]. As a consequence, wherever the thickness of the fence is so high that there is significant change on either side of the fence, this method fails to inpaint properly. This can be clearly seen in Figure 7.1, where the left image is the result of the proposed algorithm, while the right image is the result of using the Split Bregman technique. Crucial chunks of information have been missed by this method that have been faithfully restored by the proposed algorithm.

(a) **(b)**

Figure 7.1: Comparison of results using (a) the proposed algorithm, and (b) Total Variation Inpainting using Split Bregman [8]

7.2 De-fencing using Learning-Based Matting [14, 31]

Learning-based matting [31] supervised learning algorithm. It has been used for fence detection in [14]. Since it is makes use of supervised learning, it needs sample data to work. As a consequence, the biggest drawback of this method, in comparison with the proposed algorithm is that it requires manual data to be fed to it that describe the approximate location of fence and non-fence regions in the Reference and Test Images. In contrast, the proposed algorithm requires no such manual labour to achieve its task.

The rest of the method is the same as that used in the proposed algorithm. However, the superiority of the proposed algorithm lies in the fact that it can handle multiple layers in an image, while [14] did not. [14] has only shown results for one non-fence layer, and fails for multiple layers. This is shown in Figure 7.2, where the left image is the result of the proposed algorithm, while the right image is that of [14] when Pixel Shift of only the Background is used to de-fence the Reference Image.

(a) **(b)**

Figure 7.2: Comparison of results using (a) the proposed algorithm, and (b) De-fencing with Learning-Based Matting [14, 31].

Chapter 8

Precautions while Capturing Images

Chapter 8

Precautions while Capturing Images

Now that the results have been clearly shown, it is equally important to reiterate some of the assumptions that have been used while performing the experiments. While recording images using the Microsoft Kinect and during preprocessing, a couple of precautions were taken so as not to invalidate the assumptions that have been mentioned before.

8.1 Sufficient Fence Dilation

As mentioned in Section 3.3, right after the fence is extracted from a depth map and preprocessed for noise removal and region filling, it is dilated. Before going into the reasons for it, observe Figures 8.1 and 8.2, where this step was skipped.

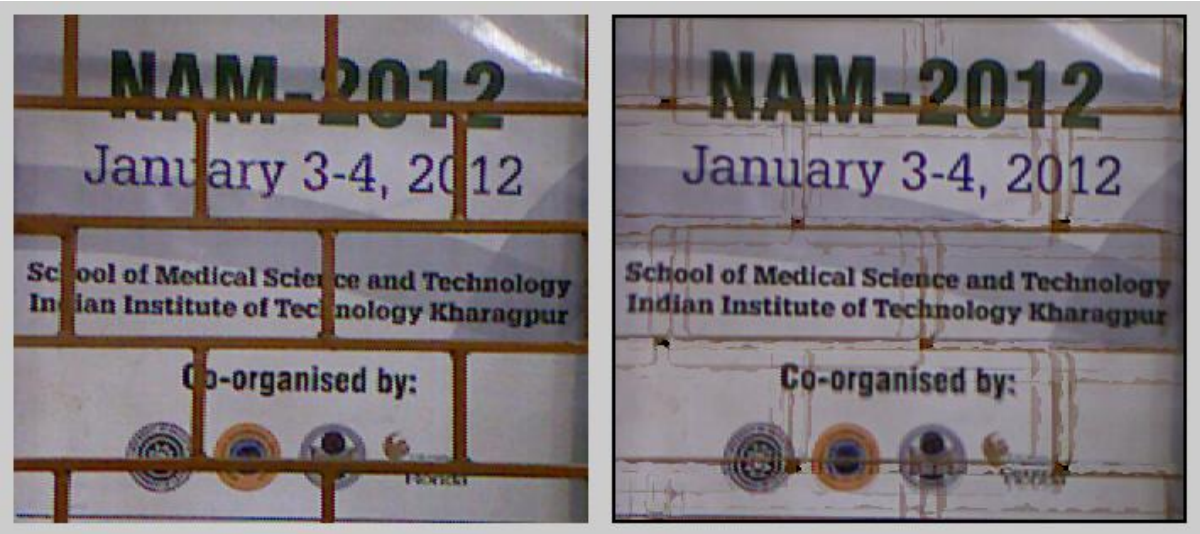


Figure 8.1: Improper de-fencing of image with rectangular fence and a poster, due to not performing fence dilation.

Figure 8.2: Improper de-fencing of image with rectangular fence and a board, due to not performing fence dilation.

As can be inferred, there are two reasons for this:

1. In the aligned depth map, there is a chance that the edges of the fences are not captured properly, due to lack of sufficient reflection of infrared light at the edges.

2. While preprocessing the image, since the fence is smoothed out and noise is removed, there is a chance that it gets eroded.

Thus, it is important to slightly dilate the fence image so that the entire fence in the colour image is covered.

8.2 Sufficient Camera Translation

As mentioned earlier in Section 4.2, while recording the images, care needs to be taken to translate the Kinect device so much so that information that was hidden in one image be revealed in another image. However, there have been cases where this precaution was not adhered to. In these cases, since the translation of the camera was not sufficient enough to

reveal information in all the pixels in the Reference Image, there are several repeated gaps in the de-fenced image. These gaps appear as black pixels in the de-fenced image. They correspond to those pixels in the Reference Image that have a portion of fence covering them in all the Test Images too.

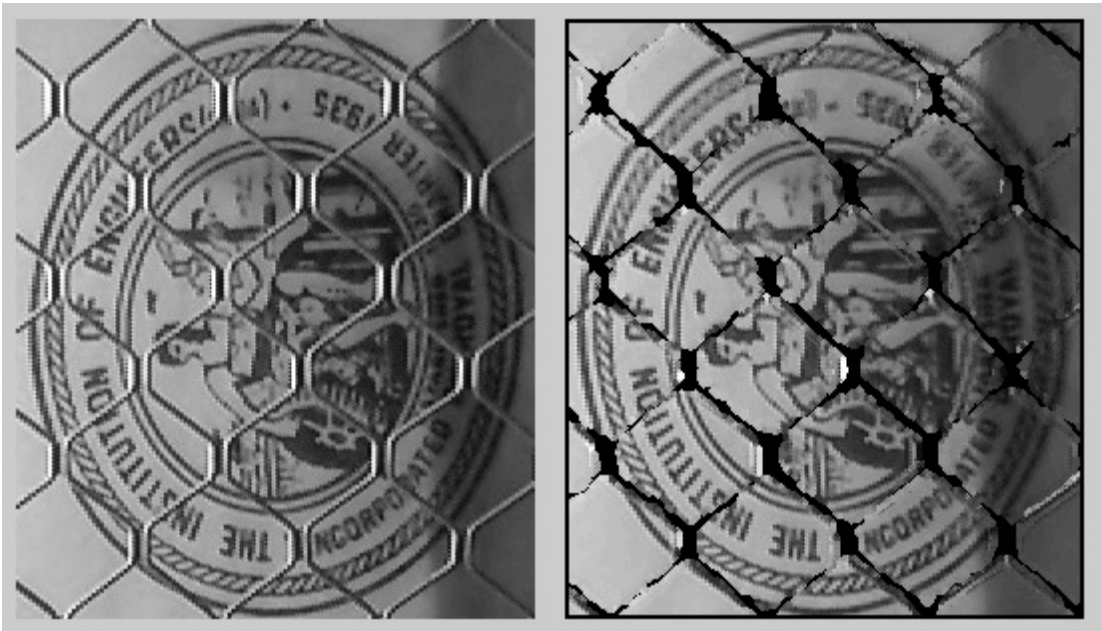


Figure 8.3: Improper de-fencing of image with diagonal fence and logo, due to insufficient camera translation.

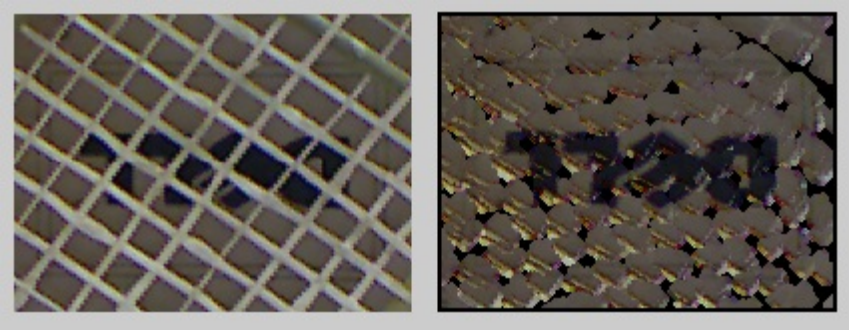


Figure 8.4: Improper de-fencing of image with diagonal bamboo fence and DELL logo, due to insufficient camera translation.

8.3 Limited pixel shift

Figures 8.5 and 8.6 display the effect of too much pixel shift between the images captured.

Figure 8.5: Improper de-fencing of image with rectangular fence and two layers, due to perspective distortion.

Figure 8.6: Improper de-fencing of image with rectangular fence and two layers, due to perspective distortion.

Pixel shift between the Reference and the Test Images being too high is a result of higher camera translations from the location used to record the Reference Image. As mentioned in Section 4.2, camera translation inherently introduces perspective distortion into the image captured, which cannot be modelled as a global Row and Column Shift. Due to this perspective distortion, when the Loopy Belief algorithm tries to combine information from multiple images, the information itself does not fit the information in the Reference Image as perfectly. Hence, the resulting de-fenced image has pixel information that is spread out over a larger area than the actuality. As a result of this, the de-fenced image appears to be blurred, or smooth, as can be seen in Figures 8.5 and 8.6.

It is important to note that such perspective distortions are present in the cases where the algorithm worked perfectly too. But in those cases, it is not as clearly visible as in Figures 8.5 and 8.6. The best way to avoid perspective distortion is to have as minimum column camera translation as possible. However, as seen in the previous section, camera translation cannot be so low as to not reveal pixel information at all. Thus, a balance needs to be maintained in the amount of translation the camera used to record images is made to go through.

Chapter 9

Conclusion

Chapter 9

Conclusion

De-fencing of images has been a very known and explored problem in computer vision. This project illustrates a new approach to solve this problem. The primary novelty of the method described is that it employs depth maps to detect fences/occlusions in scenes, as well as the multiple layers in the image, wherever applicable. It makes use of RGB-D data, recorded as a colour (RGB) image and a depth (D) map to remove occlusions from images. Data capture was performed with the help of Microsoft Kinect, and the method has been successfully tested on real-life test cases. These test cases comprised various cases depending on the kind of subjects that were chosen and the kind of fences in the scene. The method has been proven to be successful for all cases.

Various experiments were conducted with different subjects, shapes of fences and lighting conditions. Other methods that tackle the same problem were explored and the results of this method were compared, and deemed superior. Through the duration of the project, the different facets of the problem statement were discovered, explored, and solved through through extensive experimentation and research.

The results of this project have been compiled into a research paper and submitted to ACM Multimedia 2014, the 22nd ACM International Conference on Multimedia.

At the current stage, the proposed algorithm is able to handle disparity between the multiple images captured through Affine Transform. Future scope of the work is to be able to handle perspective projections between images. That shall lead to extremely robust results that shall be valid more ubiquitously and shall be useful for more prevalent applications such as in smartphones and tablets.

Bibliography

[1] Baker, S., Scharstein, D., Lewis, J., Roth S., Black, M. J., and Szeliski, R., "A database and evaluation methodology for optical flow," in *International Conference on Computer Vision (ICCV)*, 2007.

[2] Bertalmío, M., Sapiro, G., Caselles, V., and Ballester, C., "Image inpainting," *ACM Transactions on Graphics (SIGGRAPH '00), Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 417–424. http://dx.doi.org/10.1145/344779.344972

[3] Boykov, Y., and Kolmogorov, V., "An experimental comparison of min-cut/max-flow algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

[4] Chellappa, R., Jain, A., "Markov Random Fields: Theory and Applications," Academic Press, London, 1993.

[5] Criminisi, A., Perez, P., and Toyama, K., "Region filling and object removal by examplar-based image inpainting," *IEEE Transaction on Image Processing*, vol. 13,pp. 1200–1212, 2004.

[6] Cruz, L., Lucio, D., and Velho, L., "Kinect and RGBD Images: challenges and applications," in *2012 XXV SIBGRAPI Conference on Graphics, Patterns and Images Tutorials*.

[7] Forsyth, D. A., "Shape from texture without boundaries," in *Proc. European Conf. Computer Vision (ECCV)*, pp. 225–239, 2002.

[8] Getreuer, P., "Total variation inpainting using Split Bregman," *Image Processing On Line*, 2:147-157, 2012.

[9] Han, J., Shao, L., Xu, D., and Shotton, J., "Enhanced computer vision with Microsoft Kinect sensor: a review," *IEEE Transactions on Cybernetics*.

[10] Hays J., and Efros, A., "Scene completion using millions of photographs," *ACM Transactions on Graphics (SIGGRAPH 2007)*, vol. 26, no. 3, 2007.

[11] Hays, J., Leordeanu, M., Efros, A., and Liu, Y., "Discovering texture regularity as a higher-order correspondence problem," in *European Conference on Computer Vision (ECCV '06)*, 2006.

[12] Hg, R. I., Jasek, P., Rofidal, C., Nasrollahi, K., Moeslund, T. B., and Tranchet, G., "An RGB-D Database Using Microsoft's Kinect for Windows for Face Detection," *Eighth International Conference on Signal Image Technology and Internet Based Systems*, 2012.

[13] Huttenlocher, D. P., and Felzenszwalb, P. F., "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, pp. 41–54, 2006.

[14] Khasare, V., and Sahay, R., "Seeing through the fence: image de-fencing using a video sequence," *2013 IEEE International Conference on Image Processing*, Restoration and Enhancement III.

[15] Leung, T. K., and Malik, J., "Detecting, localizing and grouping repeated scene elements from an image," in *Proc. European Conf. Computer Vision (ECCV)*, pp. 546–555, 1996.

[16] Li, S. Z., *Markov Random Field modeling in image analysis*, Springer, Heidelberg, 2001.

[17] Liu, Y., Belkina, T., Hays, J., and Lublinerman, R., "Image de-fencing," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[18] Liu, Y., Collins, R., and Tsin, Y., "A computational model for periodic pattern perception based on frieze and wallpaper groups," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26(3):354–371, March 2004.

[19] Lowe, D. G., "Object recognition from local scale-invariant features," in *International Conference on Computer Vision*, Corfu, Greece, pp. 1150–1157, 1999.

[20] Morel, J. M., and Yu, G., "ASIFT, A new framework for fully affine invariant image comparison," *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[21] Park, M., Brocklehurst, K., Collins, R. T., and Liu, Y., "Image de-fencing revisited," in *Asian Conference on Computer Vision*, 2011, pp. 422–434.

[22] Park, M., Collins, R. T., Liu, Y., "Deformed lattice discovery via efficient mean-shift belief propagation," in *10th European Conference on Computer Visio*n, Marseille, France, 2008.

[23] Park, M., Collins, R. T., Liu, Y., "Deformed lattice detection in real-world images using mean-shift belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 31, no. 10, pp. 1804–1816, 2009.

[24] Rangarajan, A., Chellappa, R., "Markov Random Field models in image processing," The Handbook of Brain Theory and Neural Networks, pp. 564-567, MIT Press, 1995.

[25] Schaffalitzky, F., and Zisserman, A., "Geometric grouping of repeated elements within images," in *Shape, Contour and Grouping in Computer Vision*, pp. 165–181, 1999.

[26] Sun, D., Roth, S., and Black, M., "Secrets of optical flow estimation and their principles," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2432–2439, 2010.

[27] Sun, J., Zheng, N. N., and Shum, H. Y., "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, 2003.

[28] Tuytelaars, T., Turina, A., and Van Gool, L., "Non-combinatorial detection of regular repetitions under perspective skew," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI*, 25(4):418–432, 2003.

[29] Willsky, A. S., "Multiresolution markov models for signal and image processing," *Proceedings of the IEEE*, 90(8):1396–1458, 2002.

[30] Yu, G., and Morel, J. M., "ASIFT: An algorithm for fully affine invariant comparison," *Image Processing On Line*, 1 (2011). http://dx.doi.org/10.5201/ipol.2011.my-asift

[31] Zheng, Y., and Kambhamettu, C., "Learning based digital matting," in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 889–896.