博士論文

Video Completion via
Spatio-Temporally Consistent Motion Inpainting

（時空間整合性を利用したモーションインペインティング法によるビデオの修正）

ローハス　メナンヅロ　デラクルーズ

Video Completion via Spatio-Temporally Consistent Motion Inpainting

（時空間整合性を利用したモーションインペインティング法によるビデオの修正）

Menandro Dela Cruz Roxas

ローハス　メナンゾロ　デラクルーズ

Dissertation

Submitted in partial fulfillment of the requirement
for the degree of
Doctor of Philosophy in the field of
Information and Communication Engineering

December 2014

Video Completion via Spatio-Temporally Consistent Motion Inpainting

（時空間整合性を利用したモーションインペインティング法によるビデオの修正）

Menandro Dela Cruz Roxas

ローハス　メナンヅロ　デラクルーズ

Dissertation Committee:

Kiyoharu AIZAWA (Chair)
Shin'ichi SATO
Takeshi NAEMURA
Shunsuke KAMIJO
Takeshi OISHI

Dissertation Supervisor:

Katsushi IKEUCHI

ABSTRACT

Recent advances in the field of transportation, navigation, and virtual reality, lead to the emergence of on-vehicle cameras. Different types of cameras are mounted either on the outside, on top, on the dashboard of cars. These cameras are used to take pictures or videos of the surrounding and are used in several applications.

Virtual tours use these videos to create a virtual environment where in a person with a monitor can view and navigate. Driving simulations use such videos to create a realistic view of a certain street. Another application is the 3D reconstruction where several point of views of a certain place is necessary. Digital archiving uses these videos to document historical places.

Several issues arise from the use of street videos. The most common issue is the presence of pedestrians especially when the faces are clear enough to be recognized. Another issue is the presence of artifacts such as dead or corrupt pixels that are inherent to the camera. Some are adherent water or smudges and occluding objects on the lens of the camera or on the windshield of a car. All of these artifacts degrade the quality of the video and are problematic when used in applications that require clear view of the surrounding.

In this thesis, we address these issues by completely deleting the information (color, brightness, etc.) of the unwanted parts of the video and redrawing them using the desired values. This redrawing process is called video completion. Video completion is the process of recovering missing parts of videos either by interpolation or duplication of the known parts. The goal of video completion is a visually pleasing output video that is both spatially and temporally consistent. Spatial consistency requires objects to maintain their geometry while temporal consistency requires parts of the same object to move in the same manner.

Numerous methods have been formulated to solve the video inpainting problem. Some work directly extended image inpainting methods to videos. With the addition of a third dimension (time), existing methods result in poorly inpainted sequence, especially when both background and holes are moving. One particular approach is the use of optical flow to propagate pixels with known colors toward the hole. This approach is straight-forward, as long as the optical flow

is available in the immediately succeeding of preceding frame. However in most practical cases, such as removing pedestrians in street videos, the holes extend several frames which make immediate copying of colors using motion information insufficient. To solve this problem, one approach is to also estimate the motion inside the hole.

We present an iterative optimization approach that uses optical flow to complete the color frames. Our objective is to find the optical flow and the color of the hole such that the resulting video satisfies the requirement of the visually pleasing video.

One of our contributions is the simultaneous optical flow estimation and inpainting. By incorporating a spatially varying mask function in the data term of the optimization function, we were able to estimate the motion inside the hole by basing it to the motion at its boundary. We use two smoothness constraints to achieve coherent motion among pixels. One is the spatial smoothness constraint that is enforced by using a total variation minimization among neighboring motions. The use of spatial smoothness constraint on the motion allows us to keep the motion of the hole and the boundary in check as well as the total motion inside the hole itself.

The second constraint is the trajectory smoothness measure. We compute the optical flow among three frames and we get a forward and backward flow which we relate to each other via trajectory similarity measure. The trajectory measure or prior is calculated as the ratio of the forward and backward optical flow which is estimated by averaging many point values under the assumption that the motion is purely translational or the rotational motion is very low.

Another contribution of this work is the iterative optimization framework that allows us to use simultaneously compute the optical flow and propagate the color from known parts of the video into the hole. In order to do this, we modify the mask function used in data term the optical flow estimation function by inferring the distance of the frames of the source pixel and the hole. The source pixel is the known color where the optical flow inside the hole points. This distance increases as the frame of the source pixel moves farther away from the frame of the current inpainted hole. The distance is used to regulate the mask function. In our method, we propose a negative exponential value which decreases in value as the distance increases. Using this modified mask function allows us to use the intensity or color values of the inpainted pixels inside the hole as the optical flow estimation process runs. The error inside the hole is also regulated such that the total error inside is less than the total error outside. Our results show an increased quality of the completed video when the mask function is modified in this way.

Finally, we propose a refinement method on the inpainted motion. By estimating the boundary of objects (or motion) inside the hole, we are able to refine the boundary of the motion and therefore increase the accuracy of the

completed video. We estimate the edges of objects inside the hole by using the motion outside the hole and computing an affine transformation. We first estimate the location of the general area of the source pixels and then compute the edges in those pixels. Then, the edge is transferred to the hole via affine transformation. After estimating the edges, we then impose an edge preserving optical flow refinement by using a weighted non-local smoothness regularizer.

The generalized framework in this thesis allows for the improvements in computing time by using faster optical flow estimation techniques and graph-based color propagation methods. The framework is designed to accommodate such changes in the methodology and thus covers a wide variety of applications.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER

ONE

Introduction

1.1 Background

Recent advances in the field of transportation, navigation, and virtual reality have caused the emergence of video cameras that are used to capture an urban environment. These cameras are mounted in different places around a vehicle such that they can view the scene of particular interest. For example, cameras that are mounted on top of cars are used to document a cityscape for applications such as virtual tours, 3D modeling, digital archiving, and driving simulation. Dashboard cameras, on the other hand, are used to monitor the behavior of intermediate vehicles or the driver itself and are useful for documenting different traffic incidents.

Several issues arise from the use on-vehicle video cameras that could cause problems in certain applications. For example, the presence of pedestrians in videos that are used for virtual tours and digital archiving pose privacy issues especially when the faces are clear enough to be recognized. This issue is very common in crowded places such as tourist spots or streets. A simple and common solution to address this problem is to blur or blackout the people's faces. However, in some applications, simple blurring is not enough especially because it removes the visual appeal of the video. Oftentimes, a complete view of the facade of a building is also necessary and a complete removal of pedestrians is needed.

Another issue is the presence of artifacts such as dead or corrupt pixels that are inherent to the camera. Some are adherent water or smudges and occluding objects on the lens of the camera or on the windshield of a car. All of these artifacts degrade the quality of the video and are problematic when used in applications that require a clear view of the surrounding.

We argue that the best solution to these issues is to completely delete the information (color pixels) containing the unwanted artifacts and redraw or replace

them with the desired pixels. Although this could be done manually frame-by-frame using any image/video editing software, the process requires accuracy and time.

In this thesis, we call this process as video completion. In the succeeding section, we will define the video completion problem and its objectives and we will follow with the main contribution of this work in addressing this problem.

1.2 Video Completion Problem

Given an image sequence $\mathbf{S}$, we define the deleted region (removed pedestrians/artifacts) as the hole $\mathbf{H}$. The completion process fills in $\mathbf{H}$ with information from the known parts $\mathbf{\Omega} = \mathbf{S} \setminus \mathbf{H}$ of the sequence. The main objective of video completion is to find an $\mathbf{H}$ that makes $\mathbf{S}$ visually pleasing.

In our criteria, a visually pleasing video should have spatial and temporal consistency in both color and motion domains. We define these criteria in detail.

- **Spatially Consistent Color.** This criterion is most easily observed in static frames because it is easier to discern the geometrical appearance of an object when it is not moving. Having said that, an object that appears to be geometrically impossible (floating objects, curved building walls) is undesired. In video completion, a recognized object must satisfy its geometrical definition (i.e. a building must have doors, and windows, and its walls must be smooth.), therefore the completed parts must adhere to the original structure. Any divergence from its preconceived appearance is easily recognized by the viewer as an inconsistency.

- **Temporally Consistent Color.** This criterion suggests that if an object appears in one frame, then it should appear in all frames unless it is occluded by another object. Violation of this objective results in flickering of objects where it appears and disappears abruptly.

- **Spatially Consistent Motion.** This criterion constraints the motion of the points belonging to a same object to be smooth. Ideally, we want the motion of the hole and the boundary to be smooth such that the edge of the hole will not be apparent in the resulting video. It also suggests that the motion of the points inside the hole must be smooth.

- **Temporally Consistent Motion.** If we track a point among several frames, the motion of that point must be smooth. Although this criterion is violated in shaky videos, it still constraint the motion of a point to the general motion of the shaking, which usually comes from the camera motion.

Using these criteria, different methods can be characterized into two categories: using color frames and using motion frames in inpainting. We build our method based on the motion frames.

1.3 Thesis Contributions

This thesis addresses the problem of video completion through motion inpainting. The contributions are summarized as:

- Our first contribution is the simultaneous optical flow estimation and inpainting. By incorporating a spatially varying mask function in the data term of the optimization function, we were able to estimate the motion inside the hole by basing it to the motion at its boundary. We use two smoothness constraints to achieve coherent motion among pixels. One is the spatial smoothness constraint that is enforced by using a total variation minimization among neighboring motions. The use of spatial smoothness constraint on the motion allows us to keep the motion of the hole and the boundary in check as well as the total motion inside the hole itself. The second constraint is the trajectory smoothness measure which relates the forward and backward flow computed among three frames. The trajectory measure or prior is calculated as the ratio of the forward and backward optical flow which is estimated by averaging many point values under the assumption that the motion is purely translational or the rotational motion is very low.

- Another contribution of this work is the iterative optimization framework that allows us to use simultaneously compute the optical flow and propagate the color from known parts of the video into the hole. In order to do this, we modify the mask function used in data term the optical flow estimation function by inferring the distance of the frames of the source pixel and the hole. The source pixel is the known color where the optical flow inside the hole points. This distance increases as the frame of the source pixel moves farther away from the frame of the current inpainted hole. The distance is used to regulate the mask function. In our method, we propose a negative exponential value which decreases in value as the distance increases. Using this modified mask function allows us to use the intensity or color values of the inpainted pixels inside the hole as the optical flow estimation process runs. The error inside the hole is also regulated such that the total error inside is less than the total error outside. Our results show an increased quality of the completed video when the mask function is modified in this way.

- Finally, we propose a refinement method on the inpainted motion. By

estimating the boundary of objects (or motion) inside the hole, we are able to refine the boundary of the motion and therefore increase the accuracy of the completed video. We estimate the edges of objects inside the hole by using the motion outside the hole and computing an affine transformation. We first estimate the location of the general area of the source pixels and then compute the edges in those pixels. Then, the edge is transferred to the hole via affine transformation. After estimating the edges, we then impose an edge preserving optical flow refinement by using a weighted non-local smoothness regularizer.

1.4 Thesis Organization

This thesis is organized in a successive manner. We address the issues in video completion problem by following the criteria that we defined and propose a method and test on real and synthetic videos. We successively handles the remaining issues by discussing improvements in the proposed method.

In Chapter 2, we will review the existing methods in video completion and its immediate extensions. We will divide the methods in two categories from which we will build our proposed methods. In Chapter 3, we will discuss the very basic framework of motion inpainting using a spatially smooth motion assumption. In Chapter 4 we will then extend this approach to multiple-frames which enables us to use an additional trajectory constraint on the motion . In Chapter 5, we will define the iterative optimization framework that simultaneously estimates and inpaints the motion as well as the color frames. In Chapter 6, we will discuss the refinement method that uses estimated edge information inside the hole. Finally, we will conclude this thesis and discuss the future direction in Chapter 7.

CHAPTER

TWO

Review of Related Work

Video completion methods can be divided into two categories: using color or brightness frames and using optical flow or motion frames. Methods that use color or brightness frames rely on a similarity measure between pixels while methods that use optical flow frames rely on the similarity of motion of neighboring pixels.

2.1 Using Color Frames

Numerous methods have been formulated to solve the video completion problem. Some work directly extended image inpainting methods [Criminisi *et al.* 2003] to videos. With the addition of the time dimension, these methods result in poorly inpainted sequence especially when the background and holes are both moving.

Non-parametric sampling is the most famous video inpainting method. A global spatio-temporal optimization is proposed by [Wexler & Irani 2007]. They use 3D patches including RGB channel and intensity gradient in the horizontal and vertical directions. With the use of 3D patches the authors claim a solution in the temporal discontinuity that result from extended image inpainting techniques. However, this method suffers in both accuracy and efficiency when the hole becomes very big and the inpainted background become large.

Jia et al [Jia *et al.* 2005] propose an extension of the previous method and use large fragments based on color similarity instead of using fixed size patches and use tracking to complete the video. A large improvement in time was achieved but the quality of the inpainting is still the same. Another extension that solve the time complexity of patch matching is by Granados et al [Granados *et al.* 2012b] which allows a person to indicate locations in the video that the source of the hole might come from. In this way, the search space was dramatically reduced and the completion time was improved.

Some methods use frame alignment using features (low-rank [Zhang *et al.* 2010], SURF [Granados *et al.* 2012a], etc.) with variants such as separately inpainting background and foreground using layers [Jia *et al.* 2004]. Frame alignment only works with planar scenes and with very few visible planes. These methods also require that objects have the same size throughout the video and therefore is not applicable in most applications. Moreover, detecting planes become problematic when there are multiple planes that affect the desired value of the hole.

The most common issue with these methods is the absence of an explicit motion constraint. The success of the inpainting results depends solely on the effective comparison between neighboring pixels and the source patches. Consistent motion are somehow achieved by using 3D patches (including neighboring frames), however this approach relies too much on the presence of a periodic motion. In other words, the inpainting will only be successful if the patch has a match.

Another problem with this approach is that patches tend to diverge from consistency when the hole is too big. Even periodic motion fails because the information at the boundary is too far from the inside of the hole. In order to solve this problem, image pyramids are used which greatly improves the inpainting results.

2.2 Using Optical Flow Frames

Another approach in solving the video completion problem is the use of optical flow to propagate the pixels with known colors toward the hole. The methods that falls in this category uses two steps in completing the video. The first step is to estimate the optical flow inside the hole and then propagate the information from known parts of the video into the hole using the optical flow values. [Shiratori *et al.* 2006] complete the motion by using motion patches similar to the approach used in color frames. [Tang *et al.* 2011] also inpaint the motion but use weighted priority of patches to select the best matching patch.

Video completion can benefit from frame interpolation methods that use motion inpainting [Chen & Lorenx 2011]. The difference between the two problems is the unavailability of the spatial information in the latter. Instead of exclusively interpolating the trajectory, color and motion consistency assumption at the boundary of the hole could be used to improve the inpainting results. [M. *et al.* 2011] used optical flow to estimate the velocity of pixels between two consecutive frames and applied a TV-L1 denoising algorithm to inpaint holes. However, in their method, the solution for optical flow and inpainting are separately done.

After motion inpainting, color propagation is trivial. The success of these methods rely heavily on the accurate estimation of optical flow at the boundary of the hole. Moreover, working on the optical flow field does not ensure consistent motion between the hole and the boundary. Since most of the motion inpainting

methods use only two frames, the consistency is also limited within two frames. Aside from that, when the hole becomes very large, the motion information at the boundary will have difficulty in reaching the center of the hole.

2.3 Summary

To summarize the existing video completion methods, we present Table 2.1. Most of the effort in video completion have been focused on solving the spatial and temporal consistency in color. However, the motion characteristics of the completed video have not been sufficiently addressed. With methods that uses only color frames, the motion consistency was not addressed because there is no explicit motion constraints applied during the inpainting process. Even with the use of 3-dimensional patches for patch matching, the color channels are still not enough to sufficiently address the motion inconsistencies.

Although methods that uses motion frames implicitly address the motion consistencies, the method still lacks in several aspects. First, spatial consistency is hard to achieve if the holes become very big because the information outside the boundary of the hole could not reach the center. The convergence of global similarity measures takes longer time to a point of non-convergence. The temporal motion consistency on the other hand is hard to solve if the size of patches are small. However, increasing the patch size will result in including unnecessary information hence, a control becomes necessary.

In the next chapters, we will address the shortcomings of the methods presented here by presenting a techniques that handles all the video completion criteria into one solution.

Video Completion Objectives	Using Color Frames	Using Motion Frames (existing methods)
Spatial Consistency in Color	○	○
Temporal Consistency in Color	○	○
Spatial Consistency in Motion	× (no motion constraints)	○/× big holes have less information
Temporal Consistency in Motion	× (no motion constraints)	○/× limited by the size of patches

Table 2.1: Summary of Existing Methods

CHAPTER

THREE

Spatially Consistent (Two Frame) Motion Inpainting

Contents

One requirement of a visually pleasing video completion that has not been successfully addressed in prior works is having a spatially consistent motion. To review, spatially consistent motion means that the motion of points inside the hole moves in the same manner as the points along its boundaries. The human eye can detect very tiny inconsistencies [Smith *et al.* 2006] in motion and these are undesirable when inpainting videos. Moreover, these inconsistencies become more apparent when the background is stationary and the inpainted object is moving (i.e. walking pedestrians on the street).

Existing video inpainting methods [Wexler & Irani 2007], [Jia *et al.* 2005], [Granados *et al.* 2012b], can solve spatially consistent intensity values (differentiate from spatially consistent motion). These methods can inpaint objects that are geometrically acceptable (i.e. straight walls, smooth surface etc.) Some solutions to video inpainting are direct extension of image inpainting methods [Criminisi *et al.* 2003] with the additional temporal dimension. The time axis makes it possible to also rely on the object motion instead of just pixel values.

Spatially consistent motion can be achieved by forcing the motion inside the hole to be smooth and agrees to its boundary. Some methods that addresses this spatially consistent motion use spatio-temporal patches [Shiratori *et al.* 2006]. However, patch-based methods only works well on objects with periodic movements and suffers from over-smoothing, flickering and long computation time. Variants of this methods include separating background and foreground and us-

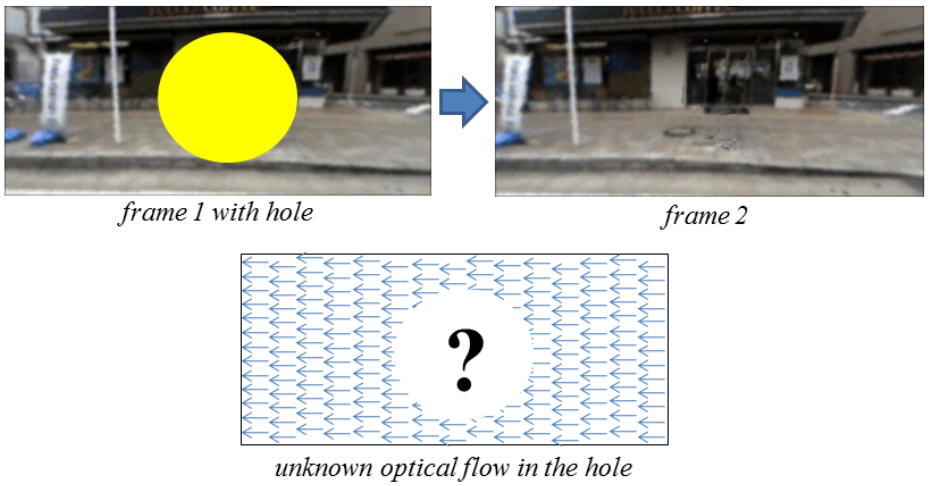

Figure 3.1: Interpolating the motion from the boundary to the hole.

ing object contours. Some of these methods directly address motion smoothness by object tracking and using motion patches instead of color.

Say for example we have two frames. If all the color in both frames are known, it is possible to solve an accurate optical flow (see Figure 3.1). However, if we introduce a hole in any of the frame, it will be impossible to know the flow of points inside it. In fact, the flow is undefined. However, we can approximate the hole motion by inferring on the relationship between it and the boundary. In this work, we assume that the motion of the boundary and the hole is smooth and therefore we can deduce the unknown motion by minimizing of the difference in motion between neighboring pixels. Then, once we have an estimate of the motion, we can propagate the color from the known parts towards the hole by following the estimated motion.

In this chapter, we present an optimization method that solves the motion inside the hole given only the optical flow values at its boundary. The overview of the method is summarized in Figure 3.2. We first describe a motion inpainting framework in Section 3.1 followed by a straightforward color propagation technique in Section 3.2. We then discuss the results and limitations in Section 3.3 and concludes the chapter in Section 3.4.

3.1 Total Variation Motion Inpainting

Given an optical flow (u, v), we impose a smoothness constraint directly on the optical flow frames. This approach solves the motion of pixels in the hole based on the motion of its boundary. For example, given a background that is planar and rigid, the equivalent motion will be uniform and therefore forces the motion

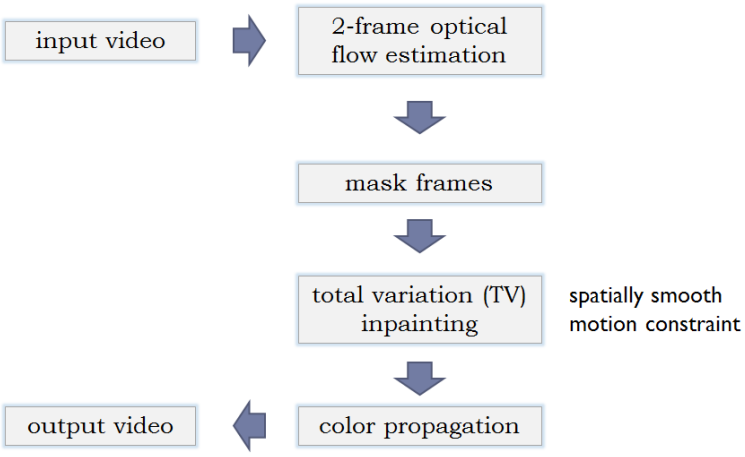

Figure 3.2: Overview of the video completion using spatally consistent motion inpainting.

in the hole to be uniform, too. Similarly, this approach also forces the motion within the hole to be smooth.

We implement the smoothness constraint by applying a total variation minimization on the optical flow per frame. This is similar to earlier work in image inpainting [Chan *et al.* 2005] [Shubhangi 2012] [Dahl *et al.* 2010] instead we apply TV on the motion domain. Note that the "per frame" implementation is justified because we do not assume that the camera motion is smooth (can be shaking and abruptly change direction or speed) at this point.

The goal of TV optimization is to minimize the following:

$$\min_{u} |\nabla \mathbf{u}|_{TV} + |\mathbf{u} - \mathbf{f}|_2^2 \tag{3.1}$$

where u is the estimated value and f is the observed variable.

The TV term is discretized as:

$$|\nabla \mathbf{u}|_{TV} = \sum_i \sqrt{(\nabla_x u)^2 + (\nabla_y u)^2} \tag{3.2}$$

where $\nabla_x u$ and $\nabla_y u$ are the motion gradient in the x and y direction, respectively.

TV regularized problems can be iteratively solved using split Bregman method as proposed by [Goldstein & Osher 2009]. TV can be solved by alternatively minimizing the following:

$$\min_{d,u} |d| + \frac{\lambda}{2} |\mathbf{u} - \mathbf{f}|_2^2 + \frac{\gamma}{2} |d - \nabla u - b|_2^2 \tag{3.3}$$

where b is the Bregman iteration variable. It is updated every iteration k such

that:

$$b^{k+1} = b^k + \nabla u - d \tag{3.4}$$

Function (3.3) is solved by alternating between d and u. The d-subproblem is given by:

$$d^{k+1} = \min_d |d| + \frac{\gamma}{2} |d - \nabla u - b|_2^2 \tag{3.5}$$

Equation (3.5) can be explicitly solved by using a generalized shrinkage formula:

$$d^{k+1} = \max\left(s^k + \frac{1}{\gamma}, 0\right) \frac{\nabla u^k + b^k}{s^k} \tag{3.6}$$

where $s^k = \sqrt{|\nabla u^k + b^k|^2}$

The u-subproblem is solved by finding the optimal u that satisfies the following equation:

$$\frac{1}{\gamma}\lambda u + \delta u = \frac{1}{\gamma}\lambda f - div\,(d - b) \tag{3.7}$$

where div is the discrete divergence and δ is the Laplacian. To use the TV optimization in inpainting, the variable λ is allowed to have spatially varying values in the image ($\lambda = 0$ in the inpainted region D and positive elsewhere). Since denoising task is not performed, the u-subproblem in this case reduces to $\delta u = div\,(d - b)$.

The u-subproblem is solved using one sweep of the Gauss-Seidel method per iteration. δ is evaluated as the 5-point Laplacian $\delta u = -4u_{i,j} + u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}$. The discrete divergence is given by $div(u) = u_{x,i-1,j} - u_{x,i,j} + u_{u,i,j-1} - u_{y,i,j}$.

To summarize the algorithm, we have:

Algorithm 1: TV motion inpainting

initialize $u, d, b = 0$
while $min > tolerance$ **do**
  solve d-subproblem
  solve u-subproblem
  update b^k
end while

3.2 Color Propagation

After completing the optical flow, the color frames are then solved. This is done by propagating the color based on the motion information. The process is

illustrated in Fig.(REF). The inpainted region is labeled as D and the known region is Ω. To solve for an unknown pixel (yellow), the optical flow information at that point is used to trace that point on to the next frame $t+1$. In the next frame, the color becomes available and therefore it is copied to the yellow square in frame t.

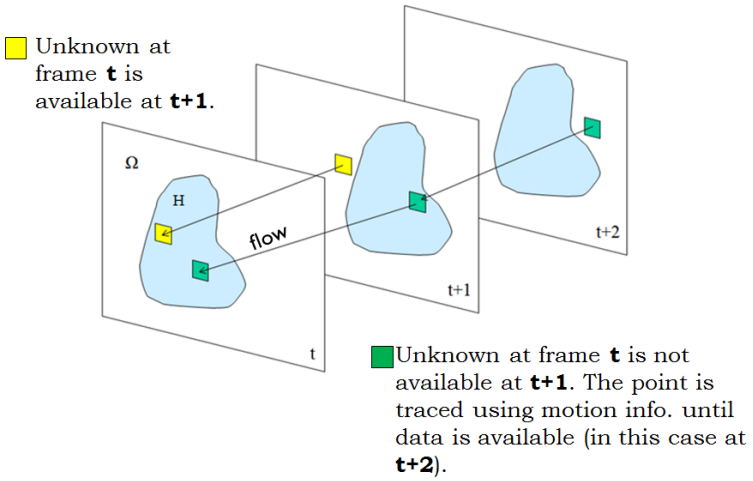

Figure 3.3: Simple color propagation. The hole color is completed by following the motion to other frames and copy the known values.

Since the hole is very large, it is possible that this simple transfer will not work. For cases where the pixel is not found on the immediate frame, the motion is followed to the succeeding frames until a known value is reached. Take for example the green pixel at frame t. To solve this problem, the pixel is traced again, following the optical flow information this time at frame $t+1$ until a known pixel is found at frame $t+2$. We then copy that color to grame $t+1$ and then to frame t. By tracing all the flows in this manner, the hole is filled completely.

It is important to note that by using this method, there is a possibility of having very few source pixels near the beginneing and end of the video especialy when the hole spans the width of the whole frames. This is the reason why we use both the forward and backward flow. However, using both flows results in inconsistency during color propagation when the flows directs to two differently colored source pixels. To solve this issue, we propagate starting from the hole boundary one pixel at a time for the whole sequence. Using this technique, we prioritize the flow which copies the color from the closest known frame and limit error building up from the propagation.

3.3 Results and Discussion

For the initial experiment, the method is tested on SINTEL [Wulff *et al.* 2012] [Butler *et al.* 2012] database called 'alley' (See Figure 3.4). This database is chosen because it contains a complete and ground truth optical flow frames. It is a computer generated 3D movie available for public use and as promotional video.

The video clip we used contains 50 frames of 1024x436 pixel resolution. The mask is manually created. First the region is estimated using the data in the optical flow. By doing a bilevel filtering, the human figure is partially extracted from the images. This partially solved mask is then manually edited to cover the entire human body. After that, the mask is dilated using a 5-point box filter to address the inconsistent intensity along the boundary.

After applying the mask on the optical flow, the inpainting method defined in the previous section is performed. The resulting frame is shown in Figure 3.4. By following the color propagation algorithm, the final result is obtained. The inpainting process takes approximately 35 seconds per frame in our MATLAB implementation on a 2.00 GHz Intel Pentium CPU with 4GB RAM.

The performance of the method is also tested on an actual road scene video (130 frames, 360x160 pixel resolution) taken from a moving vehicle. The processing time varies per frame between 30 and 50 seconds due to difference in hole size. Inpainting the whole video takes approximately 1 hour and 20 minutes. The result of inpainting is shown in Figure 3.5.

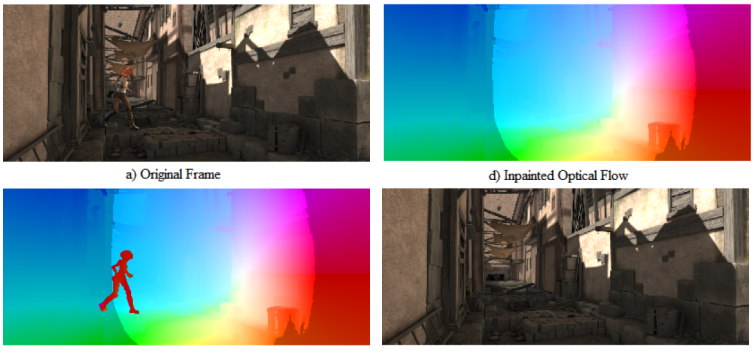

Figure 3.4: Representative frames from SINTEL database.

3.4 Chapter Summary

Maintaining spatially consistent motion is achieved by extrapolating the known motion values from the boundary into the hole using the neighborhood relationship defined by the total variation regularizer. This is done by solving the

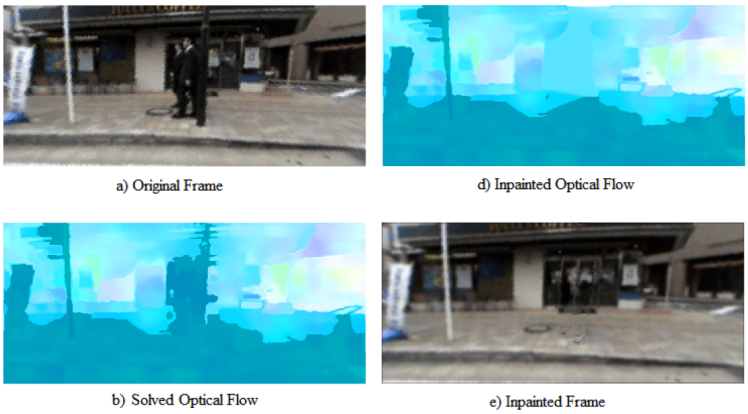

Figure 3.5: Representative frames from the road scene video.

optical flow outside of the hole using two frames. In cases where the object being inpainted is planar or have a straight edge, TV inpainting can successfully approximate its motion. Results show that by imposing this smoothness assumption, we remove the ghosting effect and allows the movement of objects to be in accordance to its surrounding.

However, street videos are often cluttered with different objects and becomes difficult to inpaint especially when the hole becomes very big. In this case, TV inpainting will fail. Moreover, since the optical flow is solved using two frames only, the motion smoothness is only limited between two frames. Since there is no explicit relationship among the image pairs, the motion consistency will be violated if the optical flow of known values fail to cohere with each other.

Needless to say, this method relies heavily on the successful estimation of the optical flow. The good news is that numerous methods have been proposed in solving the optical flow and their accuracy have improved greatly. Choosing a good estimation method is key to the success of the method proposed here.

CHAPTER

FOUR

Spatio-Temporally Consistent (Multiframe) Motion Inpainting

Contents

In the previous chapter, we have shown that two-frame motion inpainting can be done by assuming that the optical flow between the hole and its boundary is smooth. This spatial smoothness assumption, however, can only be applied to one optical flow frame.

Ideally, an object in a video could appear on all frames with an almost continuous motion (violated only by occlusions). Because of this, it is possible to

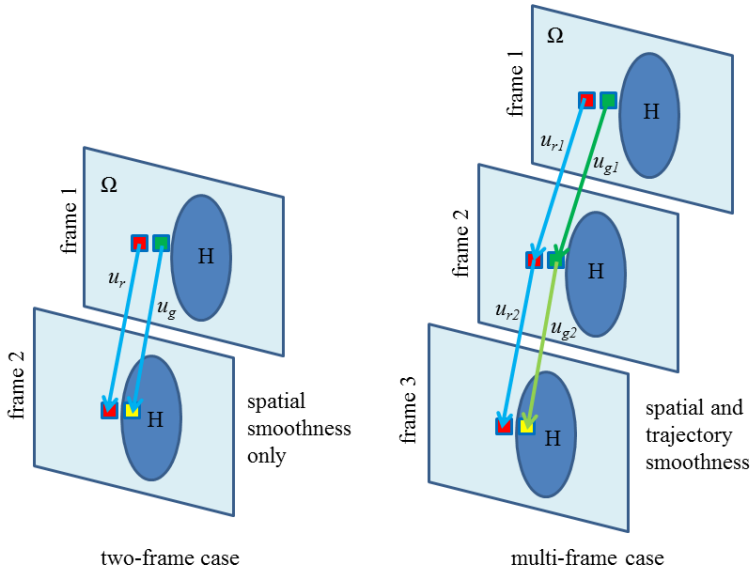

Figure 4.1: Two-frame vs. multi-frame motion inpainting.

characterize the motion of this object and therefore define the relationship among its motion as observed in the neighboring frames. During motion inpainting, we can use this idea to put additional constraints on the estimated motion of the hole and not rely exclusively on the data outside its boundary within one frame but also with that of the neighboring frames.

Take for example the motion inpainting problem in Figure 4.1. In the two-frame case, we have a known red pixel in both frames 1 and 2. We also have a known green pixel in frame 1 an unknown yellow pixel in frame 2. Since the position of the red pixel is known for both frames, we can solve for its optical flow, u_r. Following a spatially smooth motion constraint, we can assume that the green pixel will move in the same manner as the red pixel ($u_g \approx u_r$) because they are neighbors. By reasoning on the motion of the red pixel, we can say that the position of the green pixel in the second frame is indeed labeled by the yellow pixel.

Other than relying on the neighboring motion, it is safe to say that there is no other way of guessing the optical flow of the green pixel, hence the disadvantage of two-frame motion inpainting.

We will follow the same steps for the multi-frame case. In all three frames, the red pixel is present. The green pixel is present only in frames 1 and 2 and unknown in frame 3. In the same manner as the two-frame case, we can solve for the optical flow of the red pixel (u_{r1} and u_{r2}) but the green pixel can only be solved between the first and second frame (u_{g1}). By following the spatial smoothness assumption, it is safe to claim the constraint $u_{g2} \approx u_{r2}$.

In this case however, it is still possible to add another constraint on u_{g2}. If we assume that the green pixel has a constant velocity, we can also claim that $u_{g2} \approx u_{g1}$. Combining this with the spatial constraint, we can estimate u_{g2} using least squares:

$$u_{g2} = \min_{u_{g2}} (u_{g2} - u_{g1})^2 + (u_{g2} - u_{r2})^2 \tag{4.1}$$

In general cases where the velocity is not constant, we can approximate the relationship between u_{g1} and u_{g2} based on the total motion of the red pixel, with an assumption that both are rigid objects and the motion is due only to the translational motion of the camera.

Equation 4.1 can be easily extended to a continuous case, where u_{r1}, u_{r2} and u_{g1} are the known optical flow values u_Ω and u_{g2} is the optical flow inside the hole u_H. As a solution, we need to compute first u_Ω and then perform the least squares approximation in Equation 4.1 and modify it as:

$$\min_{u_H} \sum_i (u_\Omega - u_H)^2 \tag{4.2}$$

This is, in fact, a straightforward extension of the method presented in Chapter 3 with a less robust penalty function (least squares vs. total variation).

This simple approach poses two problems as is in two-frame motion inpainting.

- If the optical flow is miscalculated outside the hole, this error will obviously propagate into the hole.

- Since there are two separate optimization problem - optical flow estimation and TV (or least squares) inpainting - the computational time is also doubled.

In this chapter we will present a methodology that addresses these issues and show great improvement in estimating the motion inside the hole. In the succeeding sections, we will introduce a simultaneous optical flow estimation and inpainting method that is both spatially and temporally consistent. In order to do so, we will first extend the standard two-frame optical flow estimation framework to three frames by adding the trajectory constraint. Then we will discuss a joint optical flow estimation and inpainting method and a generalized method for multiple frames ($N > 3$) followed by the experimental results, comparisons with other methods and conclusions.

4.1 Related Work

In order to implement a multi-frame motion inpainting, we first need to be able to estimate the optical flow among several frames. In prior works, multi-frame estimation is achieved by using a temporal smoothness constraint ([Zimmer *et al.* 2011] and [Nagel 1990]) which is an assumption that the optical flow in one pixel position, say (x, y), among succeeding frames, $\{j|j = (1, 2, ..., N)\}$, is consistent. That is, $u_{x,y,j} \approx u_{x,y,j+1} \approx u_{x,y,j+N}$. This constraint, however, holds only if the pixels in two different frames belong to the same object, hence the similar motion. Nevertheless, even though temporal smoothness holds only for areas inside a single object, it has achieved better optical flow estimation results compared to using only the spatial smoothness constraint especially for constant velocity motion.

Another approach to multi-frame optical flow is the use of smooth trajectory constraint which ensures that real-world points register smooth motion among all frames. Trajectory smoothness, however, requires that points are tracked in every frame. Several works use smooth trajectory as an additional constraint to the optical flow functional. [Werlberger *et al.* 2009] solve the optical flow using three frames by imposing a hard constraint between the forward and the backward flow. In this work, since the authors assume that both motions are equal which is not always true, they reported a degradation in the result on some of their data. [Salgado & Sánchez 2007] impose a soft constraint between the flows however, their method require warping of the flows to each other, which makes it difficult to solve because flow fields refer to many different coordinate frames. [Volz *et al.* 2011] on the other hand, solve the flow fields with respect to one reference frame thus removing the need to warp them. In their method, the authors used multiple frames and only one direction which makes solving the trajectory simpler.

4.2 Three-Frame Optical Flow Estimation Framework

It is necessary to define the optical flow estimation framework in this chapter to fully explain the proposed motion inpainting method. We will revisit the widely implemented two-frame optical flow estimation proposed by [Horn & Schunck 1981] with generalized penalty functions in order to accommodate different formulations from prior works. Then, we will extend the method to three frames in Section 4.2.2 and introduce an additional trajectory constraint.

4.2.1 Horn-Schunk Model

The standard Horn-Schunck optical flow estimation function consists of an $L2$ data penalty term and the total variation (TV) term. The standard TV smoothness term allows for the optical flow to be estimated where the data penalty fails due to the absence of gradients. In motion inpainting, TV ensures that the flow field is smooth and consistent among neighboring pixels as we have discussed in Chapter 3.

We define the vector $u = (u, v)$ as the optical flow between frames I_0 and I_1, where u and v are in the x and y direction, respectively. The Horn-Schunk formulation is as follows (Eq. EQREF).

$$\min_u |I_1(x+u) - I_0(x)|_2^2 + |\nabla \mathbf{u}|_{TV} \tag{4.3}$$

Generally speaking, the $L2$ penalty function (s^2) can be replaced by ANY CONVEX FUNCTION $\psi(s)$. In literature, the common penalty functions are the L1 [Papenberg *et al.* 2006] [Wedel & T. 2008], Charbonnier [Bruhn *et al.* 2005], and Lorentzian [Black & P. 1996]. The L1-norm is non-differentiable in contrast to the Charbonnier and the Lorentzian functions which are often used in variational estimation methods. From this point, we will designate a standard form $\psi_{TV}(s)$ to represent the TV term. As a result, the generalized two-frame optical flow function can be defined as:

$$\min_u \psi\left(I_1(x+u) - I_0(x)\right) + \psi_{TV}\left(\nabla \mathbf{u}\right) \tag{4.4}$$

Given this generalized two-frame optical flow estimation function, we can now move on to define the three-frame framework that we will be using in our motion inpainting method.

4.2.2 Three-Frame Optical Flow

Using three frames, it is possible to compute two optical flows, namely forward and backward flows, that are both based on a single reference frame. In prior works [Randriantsoa & Berthoumieu 2000], the forward and backward flows are estimated using only two frames. This is done by designating the forward flow as the mapping of pixels from I_1 to I_2, and the backward flow from I_2 to I_1 (see Figure 4.2). As a result, the two flows are spatially incoherent because they are based on two different reference frames. In other words, if we look at pixel (x, y) in the forward flow, the value is not the direct opposite (negative) of the same pixel (x, y) in the backward flow.

We will differentiate the terminologies used in our method from this definition. In our work, the forward and backward flows are defined from the same reference frame, namely I_j. The forward flow is the mapping of pixels from I_j to I_{j+1}

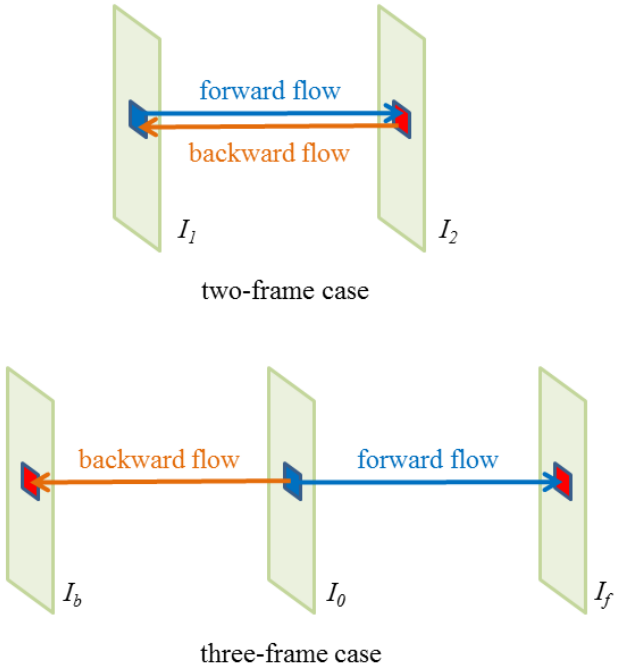

Figure 4.2: Forward and backward optical flow difference in definition between two-frame [Randriantsoa & Berthoumieu 2000] and three-frame methods.

and the backward flow is the mapping from I_j to I_{j-1}. To simplify the following definitions, we will use the subscript $\{b, 0, f\}$ instead of $\{j-1, j, j+1\}$ to represent $\{backward, reference, forward\}$ terms.

Using the function in 4.4, we can define the forward and backward flows u_f and u_b, respectively, as:

$$\min_{u_f} \psi\left(I_f(x+u_f) - I_0(x)\right) + \psi_{TV}\left(\nabla \mathbf{u_f}\right) \tag{4.5}$$

$$\min_{u_b} \psi\left(I_b(x+u_b) - I_0(x)\right) + \psi_{TV}\left(\nabla \mathbf{u_b}\right) \tag{4.6}$$

As we can see, the two functions are still independent of each other and can be minimized entirely separately. However, since both u_f and u_b are based on a single reference frame, under certain assumptions it is possible to derive a relationship between the two. For example, we can say that u_f and u_b have the same magnitude but opposite in direction (i.e., if the pixels are moving in a constant velocity), therefore we can derive the constraint $u_f + u_b = 0$. Other relationships can be derived from constrained camera models and rigid object motions. These relationships will be left for discussion in the next chapter.

For simplicity reasons, let us assume that the relationship between u_f and u_b

can be described by a general function $\phi(u_f, u_b)$. Under this assumption, we can subject 4.5 and 4.6 to the following strict constraint:

$$\phi(u_f, u_b) = 0 \tag{4.7}$$

We will call Equation 4.7 as the *trajectory constraint* and assume that ϕ is convex and differentiable. By using the concept of augmented Lagrangian method of multipliers, we must be able to combine 4.5 and 4.6 into one function by introducing a dual update variable $b^k = (b_u^k, b_v^k)$ as follows:

$$\begin{aligned}\min_{u_f, u_b} & \psi\left(I_f(x+u_f) - I_0(x)\right) + \psi_{TV}\left(\nabla \mathbf{u_f}\right) \\ & + \psi\left(I_b(x+u_b) - I_0(x)\right) + \psi_{TV}\left(\nabla \mathbf{u_b}\right) \\ & + \frac{\lambda_2}{2}\left|\phi(u_f, u_b) - b^k\right|_2^2 + const.\end{aligned} \tag{4.8}$$

The function in 4.8 can be thought of as three separate energy functions that affects different aspects of the estimation process - data fidelity and the two smoothness constraints, spatial and trajectory. The data term contributes to the correct motion when the color channel is available. It also allows for the estimated motion along the boundary of the hole to be precise which helps correctly estimating the unknown flow inside the hole. On the other hand, the spatial energy term constricts neighboring motions to be smooth and the trajectory energy term assumes object to have a smooth velocity.

It is also possible to control the effect of each terms on the resulting optical flow. We can do this by multiplying spatially dependent variables on each of the terms. This technique is particularly useful in cases where we want to limit the strength of one energy and increase the influence of another. For example, we can base the limitation in smoothness of the motion on the image gradients and say that if the color channel has a strong boundary, the motion along that boundary should have a discontinuity. This technique is not new and has been used in many work [Black & P. 1991] [Li *et al.* 2012] among others.

Specifically, in the motion inpainting process, we can completely remove the effect of color inside the hole by assigning a binary function to label the hole (0) and the known region (1) and multiply it to the data term. It is also possible to assign a function based on the accuracy of the color inside the hole after the completion process in order to refine the estimated flow (see Chapter 5 and 6). As a summary, the three energy terms are given by:

$$E_{data}(u_f, u_b) = \lambda_d \left[\psi\left(I_f(x+u_f) - I_0(x)\right) + \psi\left(I_b(x+u_b) - I_0(x)\right)\right] \quad (4.9)$$

$$E_{spatial}(u_f, u_b) = \lambda_s \left[\psi_{TV}\left(\nabla \mathbf{u_f}\right) + \psi_{TV}\left(\nabla \mathbf{u_b}\right)\right] \quad (4.10)$$

$$E_{trajectory}(u_f, u_b) = \frac{\lambda_t}{2}\left|\phi(u_f, u_b) - b^k\right|_2^2 + const. \quad (4.11)$$

with λ_d, λ_s and λ_t as the spatially dependent variables. For visual simplicity, the three-frame optical flow estimation function can be represented by:

$$\min_{u_f, u_b} E_{data}(u_f, u_b) + E_{spatial}(u_f, u_b) + E_{trajectory}(u_f, u_b) \quad (4.12)$$

4.3 Joint Optical Flow Estimation and Inpainting

To make this paper self-sufficient, we describe the solution to (4.12) in detail and summarize in Algorithm 2.

4.3.1 Linearization of the Data Penalty Term

We first need to make the inner terms of the data energy in Equation 4.9 differentiable. To do this, we linearize the image I_f using the first order Taylor approximation near $x + u_f$. This yields:

$$I_f(x+u_f, y+v_f) = I_f(x,y) + u_f \frac{dI_f}{dx} + v_f \frac{dI_f}{dy} \quad (4.13)$$

The inner data term for I_f therefore becomes:

$$\psi\left(I_f(x+u_f, y+v_f) - I_0(x,y)\right) = \psi\left(u_f I_{f_x} + v_f I_{f_y} + I_{f_t}\right) \quad (4.14)$$

where I_{f_x} and I_{f_y} are the partial derivatives of I_f in the x and y directions and $I_{f_t} = I_f(x,y) - I_0(x,y)$

We do the same for I_b and the whole data term can be rewritten as:

$$E_{data}(u_f, u_b) = \lambda_d \left[\psi\left(u_f I_{f_x} + v_f I_{f_y} + I_{f_t}\right) + \psi\left(u_b I_{b_x} + v_b I_{b_y} + I_{b_t}\right)\right] \quad (4.15)$$

By linearizing E_{data}, the term is now differentiable and can be minimized using variational techniques.

4.3.2 Continuous Refinement and Bicubic Warping

In practice, the linearization in Equation 4.15 will not be satisfied because the image gradients will have large variations between I_f and I_0 especially when the motions are large. This will result in a failed estimation of the optical flows. A famous method to address this issue is to use image pyramids [Burt 1981] to make the images smaller and hence the estimated motion lesser. Using a smaller image, the optical flow is solved and is used as an initial value for the larger image. Another approach is to solve for a sparse optical flow using patch matching [Bao *et al.* 2014] or point correspondences of invariant features such as SIFT [Li *et al.* 2012].

Assuming that u_f is already a close approximation of the desired value, we warp the I_f using this u_f and then solve for the differential $\Delta u_f = (\delta u_f, \delta v_f)$ instead. The new warped image is given by $\bar{I_f}(\Delta u_f) = I_f(x + u_f + \Delta u_f)$. We then rewrite the data term again using the warped image for the forward flow as:

$$E_{data}(u_f) = \lambda_d \left[\psi \left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right)\right] \tag{4.16}$$

We then minimize the energy in terms of Δu_f as:

$$\min_{\delta u_f} \lambda_d \left[\psi \left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right)\right] \tag{4.17}$$

In other words, an initial guess of u_f is refined by solving the optimization function within Δu_f. This step is often called continuous refinement.

To use the above refinement technique, we first need to warp image I_f to the reference frame using the initial value of u_f. The optical flow (u_f, v_f) maps the pixels of the warped $\bar{I_f}$ from its source values in I_f (see Figure 4.3). A pixel in $\bar{I_f}$, $(\bar{x}, \bar{y})$ will obviously point to an inexact location in I_f. To estimate the value of while considering all the four points, we follow the *bicubic interpolation* method. It is an extension of the cubic interpolation to two-dimensional domain. We use this type over the bilinear one because it results in a smoother warping that is closer to the original image. Figure 4.4 shows an example of the warped image $I_{f-warped}$ using the optical flow u_f and the residual with the reference frame I_0.

4.3.3 Solving the Image Gradients

After warping the images, we now solve for the image derivatives I_{f_x} and I_{f_y}. The continuous image derivatives can be approximated by solving the discrete image gradients. We do this by convolving the image with a kernel filter given by Equation 4.18 in both x and y directions. The image gradients are shown in Figure 4.5.

$$K = \frac{1}{12} \begin{bmatrix} 1 & -8 & 0 & 8 & -1 \end{bmatrix} \tag{4.18}$$

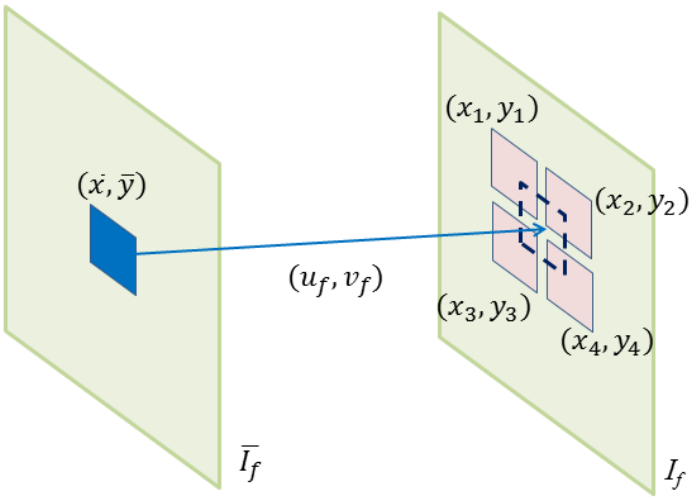

Figure 4.3: Mapping of the pixel in the warped image using the optical flow points to an inexact location in the source frame. The problem is solved using bicubic interpolation.

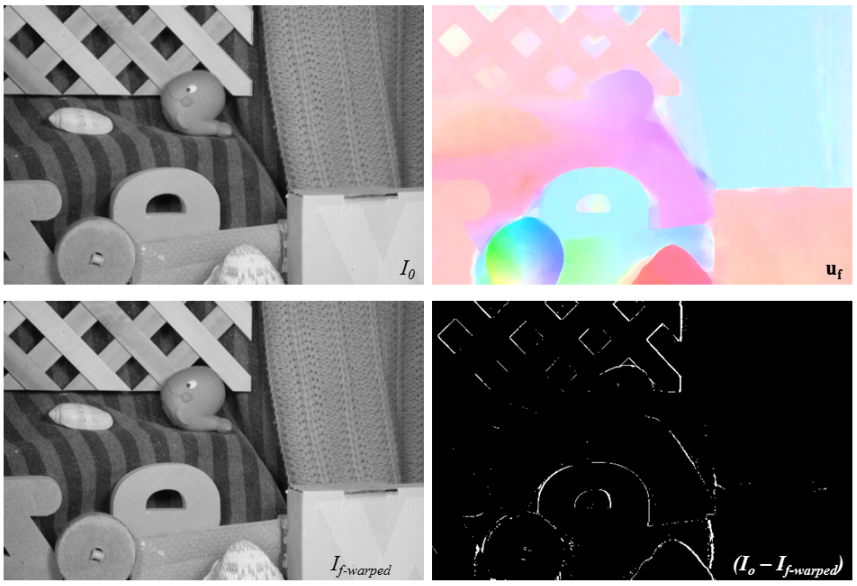

Figure 4.4: Warping of frame I_f to the reference frame I_0 using the optical flow u_f yields $I_{f-warped}$. The bottom right image shows the residual between the reference frame and the warped image. Notice the error localized along the motion boundaries.

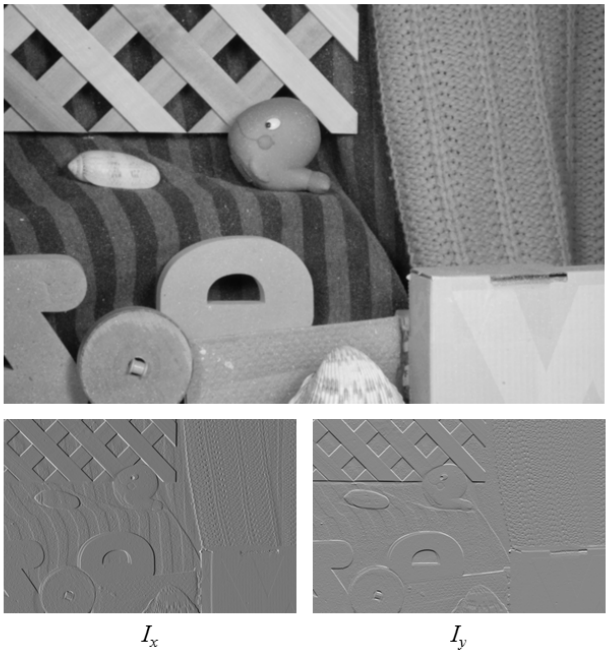

Figure 4.5: Image gradients I_x and I_y of the top image in the x and y directions, respectively.

Since we are assuming brightness constancy, we can say that the image gradient of the warped I_f is equal to I_0. However, this is not always the case due to non-uniformity of illumination, noise, and arbitrary brightness variations. To compensate on these changes, the image gradients of both image is averaged:

$$\bar{I_{f_x}} = \frac{1}{2}(I_{f_x} + I_{0_x}) \tag{4.19}$$

We do the same thing in the y component of the images. The time derivative I_{f_t} remains as the difference $I_f(x, y) - I_0(x, y)$.

4.3.4 Hole Warping

In this section, we will discuss the need to warp not only the images but also the hole (or mask frames) (λ_d). Take a look at the example optical flow estimation in Figure 4.6. In this illustration, we are estimating the flow u_f between frames I_0 and I_f. We warp I_f using an initial estimate u_{f0} to get $\bar{I}_f$. We then overlay $\bar{I}_f$ with a warped mask and non-warped mask (hole labeled by a dashed line).

In the case where we did not warp the hole, notice that the supposedly removed object h was existing in $\bar{I}_f$. In contrast, the removed object is completely covered by the mask if we indeed warp the hole. As a result of this, the optical flow of the case without hole warping yield more error in the results (labeled x)

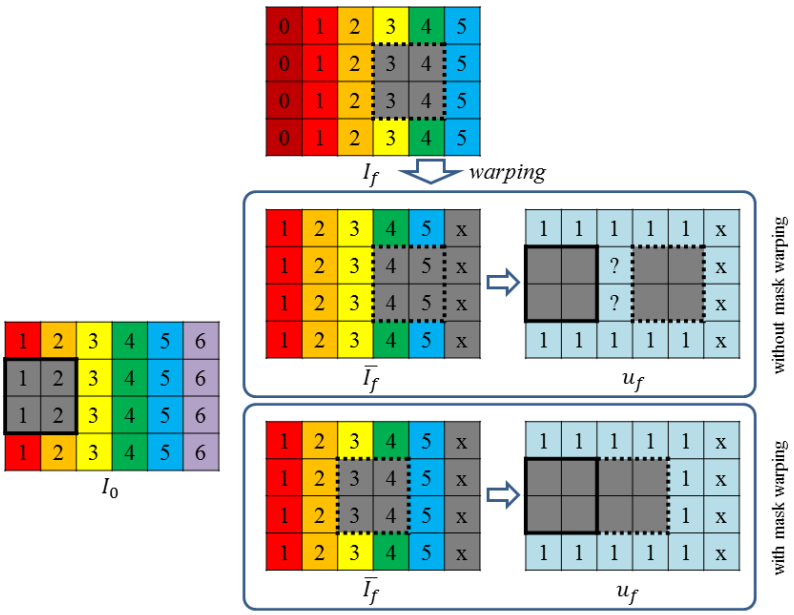

Figure 4.6: Effect of mask warping on the estimated flow with the mask in dashed lines. Removing the pixels labeled h, the non-warped hole yields more miscalculated optical flow labeled as x.

compared to the one where we warped the hole.

We show an example of the improvement in the optical flow estimation in Figure 4.7. In this sequence, we introduce a hole that moves from right to left. We tested the motion inpainting method with and without warping the hole. The results show that not only does non-warped hole creates error in excess regions (parts not masked at the warped image), but also the error seems to propagate into the hole, thus degrading the whole inpainting process. In this example, the maximum end-point difference is 0.6720.

4.3.5 Alternating Direction Method of Multipliers

The next step in the joint estimation and inpainting is to numerically solve the optimization function. We do this by using the augmented Lagrange method of multipliers, also known as alternating direction method. Combining the linearized data term, the warped images and hole and the image gradients, we want

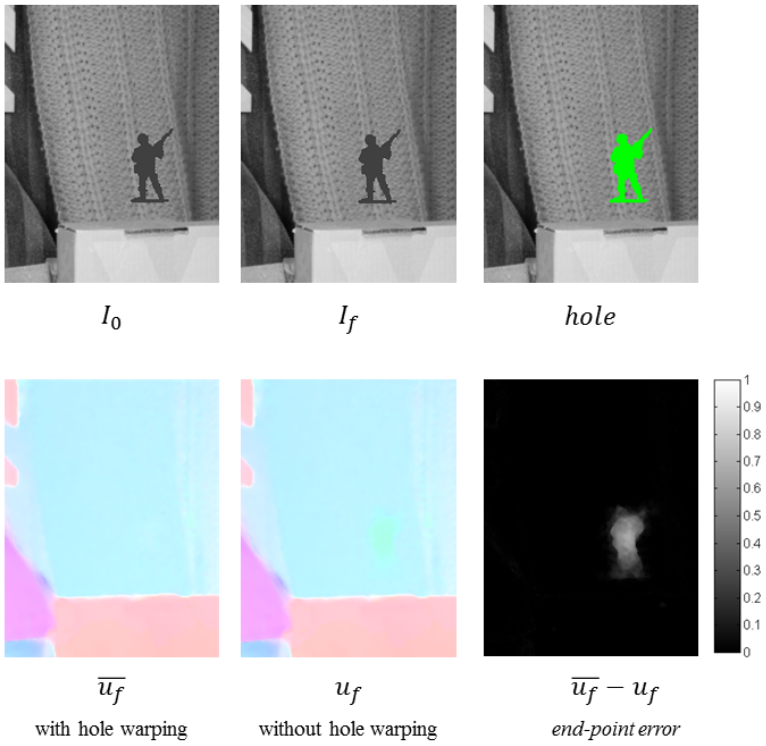

Figure 4.7: Effect of hole warping on the inpainted motion in the hole.

to minimize Function 4.20 for Δu_f and Δu_b.

$$\begin{aligned}\min_{\Delta u_f, \Delta u_b} \lambda_d \left[\psi\left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right) + \psi\left(\delta u_f \bar{I_{b_x}} + \delta v_f \bar{I_{b_y}} + \bar{I_{b_t}}\right)\right] \\ + \lambda_s \left[\psi_{TV}\left(\nabla(u_f + \Delta u_f)\right) + \psi_{TV}\left(\nabla(u_b + \Delta u_b)\right)\right] \\ + \frac{\lambda_t}{2}\left|\phi(u_f + \Delta u_f, u_b + \Delta u_b) - b^k\right|_2^2 + const. \end{aligned} \tag{4.20}$$

We first solve for u_f by holding u_b and b^k constant, then solve for u_b with u_f and b^k constant. After that, we update the dual variable b^k. The alternating

direction method is:

$$
\begin{aligned}
u_b^k = const. : u_f^{k+1} &= \min_{\Delta u_f} \lambda_d \psi \left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right) + \lambda_s \psi_{TV} \left(\nabla(u_f + \Delta u_f)\right) \\
&\quad + \frac{\lambda_t}{2} \left|\phi(u_f + \Delta u_f, u_b + \Delta u_b) - b^k\right|_2^2 \\
u_f^k = const. : u_b^{k+1} &= \min_{\Delta u_b} \lambda_d \psi \left(\delta u_b \bar{I_{b_x}} + \delta v_b \bar{I_{b_y}} + \bar{I_{b_t}}\right) + \lambda_s \psi_{TV} \left(\nabla(u_b + \Delta u_b)\right) \\
&\quad + \frac{\lambda_t}{2} \left|\phi(u_f + \Delta u_f, u_b + \Delta u_b) - b^k\right|_2^2 \\
b^{k+1} &= \phi(u_f + \Delta u_f, u_b + \Delta u_b) - b^k
\end{aligned} \tag{4.21}
$$

The u_f and u_b sub-problem can be solved in the same manner. We do this by solving for the Euler-Lagrange equations and finding the expression for Δu_f and Δu_b. For this implementation, we use the Charbonnier penalty ($\epsilon = 0.001$), and define a scaling coefficient β_d and β_s such that:

$$
\begin{aligned}
\beta_d &= \frac{1}{\sqrt{\left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right)^2 + \epsilon^2}} \\
\beta_s &= \frac{1}{\sqrt{(\nabla u_f)^2 + (\nabla v_f)^2}}
\end{aligned} \tag{4.22}
$$

The Euler-Lagrange equations for u_f become:

$$
\begin{aligned}
0 &= \lambda_d \beta_d I_{f_x} \left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right) + \lambda_s \beta_s \nabla^2 \left(u_f + \delta u_f\right) + \lambda_t \psi'(\delta u_f) \\
0 &= \lambda_d \beta_d I_{f_y} \left(\delta u_f \bar{I_{f_x}} + \delta v_f \bar{I_{f_y}} + \bar{I_{f_t}}\right) + \lambda_s \beta_s \nabla^2 \left(v_f + \delta v_f\right) + \lambda_t \psi'(\delta v_f)
\end{aligned} \tag{4.23}
$$

For the TV term, we use the approximation of the Laplacian:

$$
\nabla^2 u = u - \bar{u} \tag{4.24}
$$

where $\bar{u}$ is obtained using a 4-point box filter $\bar{u} = 0.25(u_{x+1,y} + u_{x-1,y} + u_{x,y+1} + u_{x,y-1})$.

As an example, if we assume a constant velocity motion, the trajectory constraint is given by $\psi(\delta u_f) = (u_f - u_b + \Delta u_f - \Delta u_b)^2$. We substitute the Laplacian with the approximation above and rewrite the Euler-Lagrange equations in 4.23 can be as a linear equation in the form:

$$
A \begin{bmatrix} \delta u_f \\ \delta v_f \end{bmatrix} = B \tag{4.25}
$$

where

$$
A = \begin{bmatrix} \lambda_d \beta_d I_{f_x}^2 + \lambda_s \beta_s + \lambda_t & \lambda_d \beta_d I_{f_x} I_{f_y} \\ \lambda_d \beta_d I_{f_x} I_{f_y} & \lambda_d \beta_d I_{f_y}^2 + \lambda_s \beta_s + \lambda_t \end{bmatrix} \tag{4.26}
$$

and

$$B = \begin{bmatrix} \lambda_s\beta_s\delta\bar{u}_f - (\lambda_t + \lambda_s\beta_s)u_f + \lambda_s\beta_s\bar{u_f} - \lambda_t u_b - \lambda_d\beta_d I_{f_x} I_{f_t} \\ \lambda_s\beta_s\delta\bar{v}_f - (\lambda_t + \lambda_s\beta_s)v_f + \lambda_s\beta_s\bar{v_f} - \lambda_t v_b - \lambda_d\beta_d I_{f_y} I_{f_t} \end{bmatrix} \tag{4.27}$$

After we solve for $\begin{bmatrix} \delta u_f & \delta v_f \end{bmatrix}^T$ we update the actual flow ${u_f}^{k+1} = {u_f}^k + \Delta u_f$. There is no need to allow the solution to converge at this point, because any progress that we make will be compensated by the dual update variable, thus we only run the sub-problems once. To reduce the errors and to speed up the convergence, we also perform a median filter after every sub-problem using a 5x5 neighborhood. We follow the same process for u_b and then update the dual variable b_{k+1}.

Algorithm 2: Inner Iteration

Require: u_f, u_b;
  initialize $u_f, u_b, b^0, k \leftarrow 0$
  while convergence$\neq$TRUE **do**
    linearize I_f, I_b
    Hole Warping
    $u_b = constant$, solve u_f
    $u_f = \text{median}(u_f)$
    $u_f = constant$, solve u_b
    $u_b = \text{median}(u_b)$
    update b^{k+1}
    $k \leftarrow k + 1$
  end while

4.4 Extension to Multiple Frames

In the previous section, we have described a three-frame optical flow estimation and inpainting using only three frames. It is logical to say that we can extend the method we described to multiple frames and modify the constraints to accommodate all the included frames. Luckily, the data and spatial terms do not change at all and are independent among all the frames. The additional burden goes to characterizing the motion of the objects and defining a new trajectory constraint.

We can limit the coverage of the multiple frame case if we allow the pairing of optical flows based on the distance of the frame to the reference ($j + 1$ and $j - 1$, $j + 2$ and $j - 2$, etc.) We then extend three-frame technique using these pairs.

The extension of the three-frame motion inpainting framework to multiple frames is not very straightforward. If we redefine function 4.12 to multi-frame, we have:

$$\min_{u_{j+n}} \sum_{n=-N}^{N} E_{data}(u_{j+n}) + E_{spatial}(u_{j+n}) + E_{trajectory}(u_{j+n}) \tag{4.28}$$

where N is the number of optical flow pairs that we want to compute (i.e., 1 for three frames, 2 for five frames and so on...). By following the solution in Section 4.3.5, Function 4.28 can be minimized by alternately solving:

$$\begin{aligned}
u_{j-n}^{k} = const. : u_{j+n}^{k+1} = & \min_{\Delta u_f} \lambda_d \psi \left(\delta u_{j+n} \bar{I_{j+n_x}} + \delta v_{j+n} \bar{I_{j+n_y}} + \bar{I_{j+n_t}} \right) + \lambda_s \psi_{TV} \left(\nabla (u_{j+n} + \Delta u_{j+n}) \right) \\
& + \frac{\lambda_t}{2} \left| \phi(u_{j+n} + \Delta u_{j+n}, u_{j-n} + \Delta u_{j-n}) - b^k \right|_2^2 \\
u_{f}^{k} = const. : u_{j-n}^{k+1} = & \min_{\Delta u_{j-n}} \lambda_d \psi \left(\delta u_{j-n} \bar{I_{j-n_x}} + \delta v_{j-n} \bar{I_{j-n_y}} + \bar{I_{j-n_t}} \right) + \lambda_s \psi_{TV} \left(\nabla (u_{j-n} + \Delta u_{j-n}) \right) \\
& + \frac{\lambda_t}{2} \left| \phi(u_{j+n} + \Delta u_{j+n}, u_{j-n} + \Delta u_{j-n}) - b^k \right|_2^2 \\
b^{k+1} = & \phi(u_{j+n} + \Delta u_{j+n}, u_{j-n} + \Delta u_{j-n}) - b^k
\end{aligned} \tag{4.29}$$

Assuming that the velocity of all points in every frame is constant, the dual update is given by:

$$b^{k+1} = u_{j+n} + u_{j-n} + b^k \tag{4.30}$$

However, the convergence of 4.29 is difficult to prove. In fact, as of this writing, there is no literature that have proven the convergence of 4.29 without certain assumptions [Chen *et al.* 2014] [Lin *et al.* 2014]. Luckily, it is possible ensure convergence if we can prove the orthogonality of any $N-1$ pair of the coefficient matrices in 4.30.

It is possible to modify the trajectory constraint such that the resulting augmented Lagrangian will converge. First, we will show a simple example using five frames and constant velocity assumptions. Again, we will use a simpler notation $\{bb, b, 0, f, ff\}$ to represent subscripts $\{j-2, j-1, j, j+1, j+2\}$. The optical flows that we want to estimate are u_{bb},u_b,u_f, and u_{ff}.

First, we let:

$$F(u) = E_{data}(u) + E_{spatial}(u) \tag{4.31}$$

We also regard $u_1 = (u_f, u_b)$ and $u_2 = (u_{ff}, u_{bb})$. Then we can rewrite 4.28 as:

$$\min_{u_1, u_2} F(u_1) + F(u_2) + E_{trajectory}(u_1, u_2) \tag{4.32}$$

where $E_{trajectory}(u_1, u_2) = \frac{\lambda_t}{2} \left|2u_1 - u_2 - b_{12}^k\right|_2^2$. The augmented Lagrangian method simply becomes:

$$u_1^{k+1} = \min_{u_1} F(u_1) + \frac{\lambda_t}{2} \left|2{u_1}^k - {u_2}^k - b_{12}^k\right|_2^2 \tag{4.33}$$

$$u_2^{k+1} = \min_{u_2} F(u_2) + \frac{\lambda_t}{2} \left|2{u_1}^{k+1} - {u_2}^k - b_{12}^k\right|_2^2 \tag{4.34}$$

$$b^{k+1} = 2{u_1}^{k+1} - {u_2}^k - b_{12}^k \tag{4.35}$$

Obviously, the solutions to 4.33 and 4.34 take the same process. For 4.33, we substitute u_f and u_b for u_1. If we elaborate the function, we can see:

$$\min_{u_f, u_b} F(u_f) + F(u_b) + \frac{\lambda_t}{2} \left|2(u_f, u_b)^k - {u_2}^k - b_{12}^k\right|_2^2 \tag{4.36}$$

subject to $u_f + u_b = 0$. Again, we will perform another ROUND of augmented Lagrangian method. To do this, we have to separate the forward and backward variables as was done in 4.8. We alternatively solve:

$$\begin{aligned} \min_{u_f, u_b} F(u_f) + F(u_b) + \frac{\lambda_t}{2} \left|2(u_f, u_b) - u_2 - b_{12}^k\right|_2^2 \\ + \frac{\lambda_t}{2} \left|u_f + u_b - b_{fb}^{\hat{k}}\right|_2^2 \end{aligned} \tag{4.37}$$

The alternating variable method then becomes:

$$u_f^{k+1} = \min_{u_f} F(u_f) + \frac{\lambda_t}{2} \left|2(u_f, u_b)^k - {u_2}^k - b_{12}^k\right|_2^2 + \frac{\lambda_t}{2} \left|{u_f}^k + {u_b}^k - b_{bf}^k\right|_2^2 \tag{4.38}$$

$$u_b^{k+1} = \min_{u_b} F(u_b) + \frac{\lambda_t}{2} \left|2(u_f, u_b)^{k+1} - {u_2}^k - b_{12}^k\right|_2^2 + \frac{\lambda_t}{2} \left|{u_f}^{k+1} + {u_b}^k - b_{bf}^k\right|_2^2 \tag{4.39}$$

$$b_{fb}^{k+1} = {u_f}^{k+1} + {u_b}^{k+1} - b_{fb}^k \tag{4.40}$$

It may not be ideal to use more than two pairs of optical flows (five frames) especially if the motion is really large. The distance between the farthest frame to the reference frame might result in larger and more occlusion regions. However, it is beneficial when the motion is slow, because we will have more areas covered for more accurate estimation along the boundary. Besides, using more frames will result in longer processing time though, the increase in time is linear in n, $\mathcal{O}(n)$.

4.5 Experimental Results

4.5.1 Comparison with TV Inpainting

We tested the motion inpainting result on Middlebury datasets with ground truth optical flow. We first introduce holes in critical parts of the images where the method might fail. We present the results of the rubberwhale, hydrangea and grove2 in Figure 4.8. We compare the results both qualitatively and quantitatively between the TV inpainting and the method described in this chapter. In all of the datasets, we were able to produce more detailed results in the inpainted motion. We also test the datasets grove3, urban2 and urban3 and show the inpainting result in Figure 4.9.

We then calculate the difference between the ground truth optical flow and the results and show the endpoint error map in Figures 4.10 and 4.11. In all the results, we show that the spatio-temporal inpainting was able to achieve more accurate inpainted motion. The error however are noticeable in areas around the object boundaries.

We then plot the endpoint error (EPE) and average angular error (AAE) of all the dataset and show the results in Figure 4.12. Aside from the urban3 dataset, the spatio-temporal inpainting has achieved better results based on the average angular error with the ground truth.

4.5.2 Comparison with Prior Work

We compare the result of the motion inpainting with our implementation of the method described in [Shiratori *et al.* 2006]. We tested the prior method with the rubberwhale and hydrangea dataset with introduced hole as in the previous section. We the solved the endpoint error between the results and the groundtruth.

Our method was able to produce more accurate results compared to [Shiratori *et al.* 2006].

4.6 Chapter Summary

The use of three frames to utilize trajectory information is effective in improving the estimated motion. Also using the masking function during motion estimation improves the estimated motion at the boundaries of objects. We showed in our results the effectiveness of our method and tested them on a database to compare with the ground truth results. We also compared our method with an existing work and showed that our method works better.

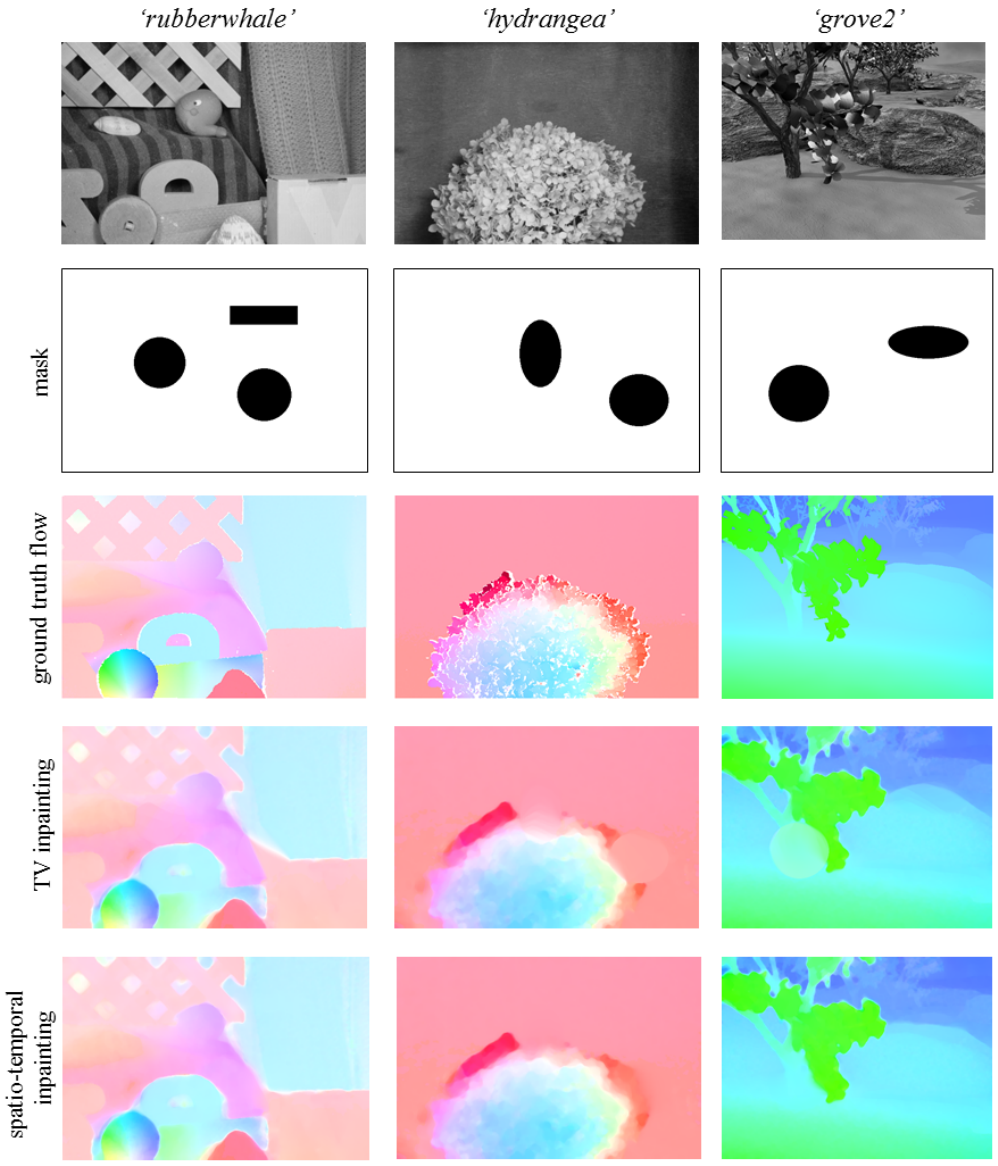

Figure 4.8: Results of motion inpainting of rubberwhale, hydrangea, and grove2 dataset with introduced hole.

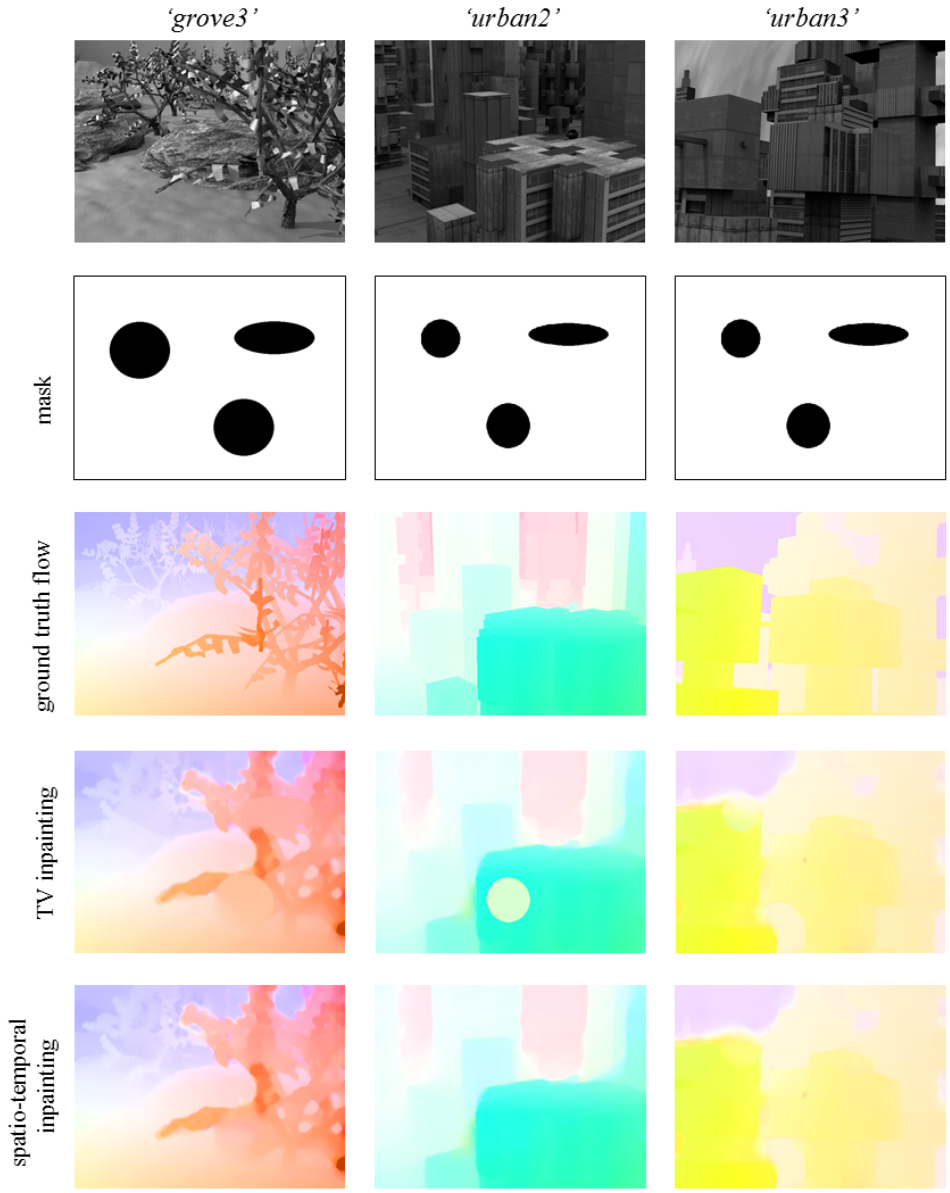

Figure 4.9: Results of motion inpainting of grove3, urban2, and urban3 dataset with introduced hole.

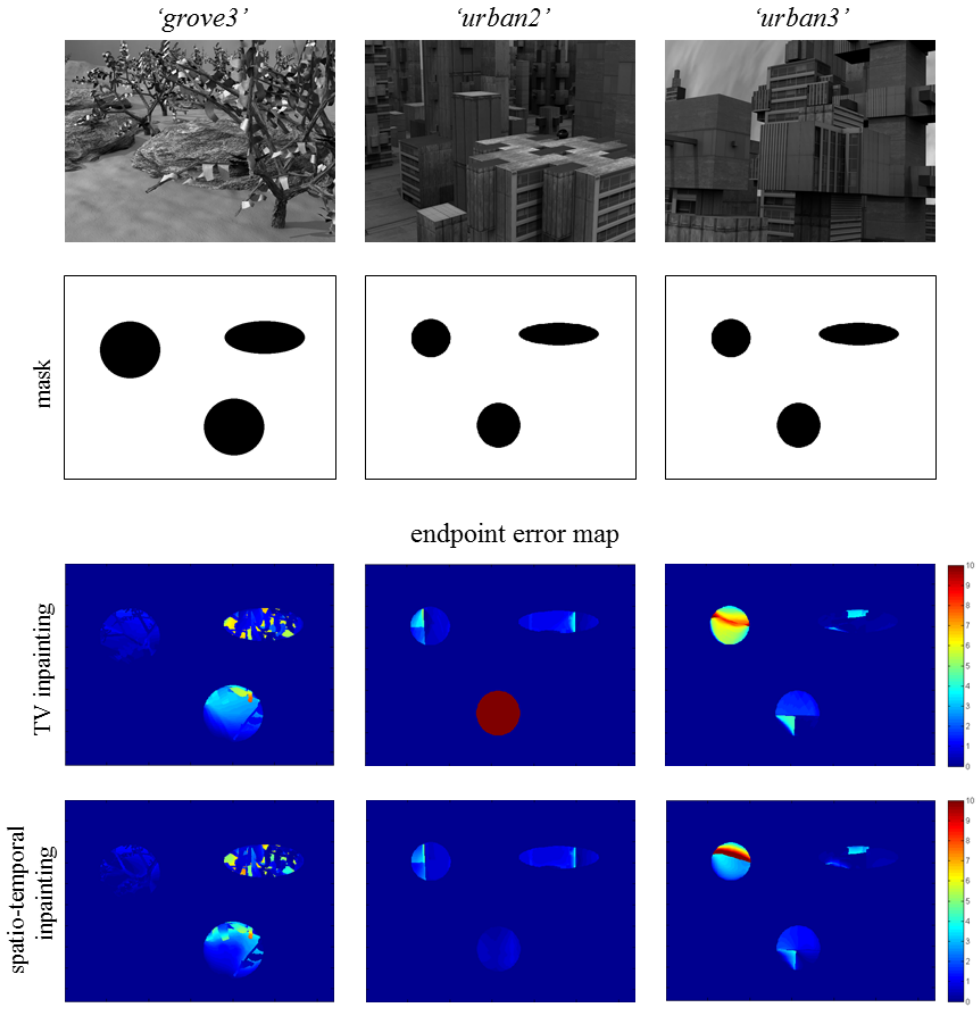

Figure 4.10: Endpoint error map of motion inpainting of rubberwhale, hydrangea, and grove2 dataset with introduced hole.

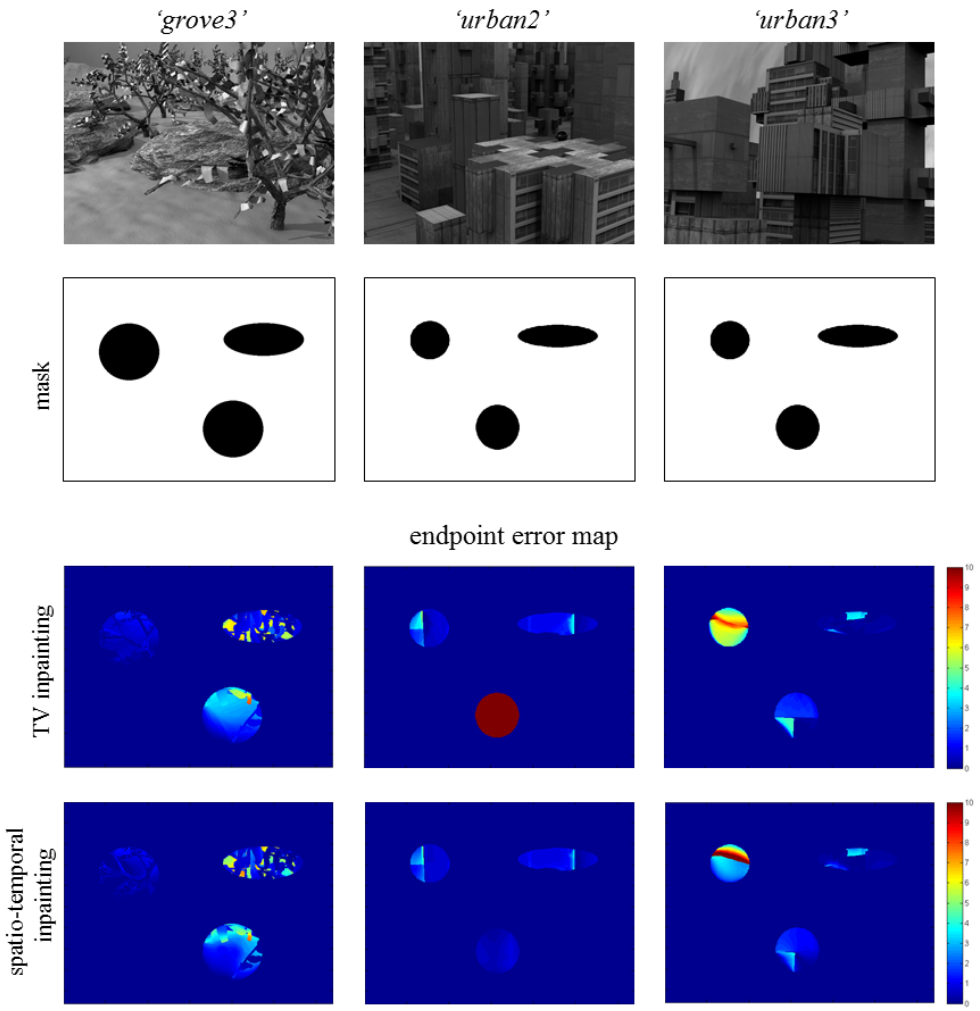

Figure 4.11: Endpoint error map of motion inpainting of grove3, urban2, and urban3 dataset with introduced hole.

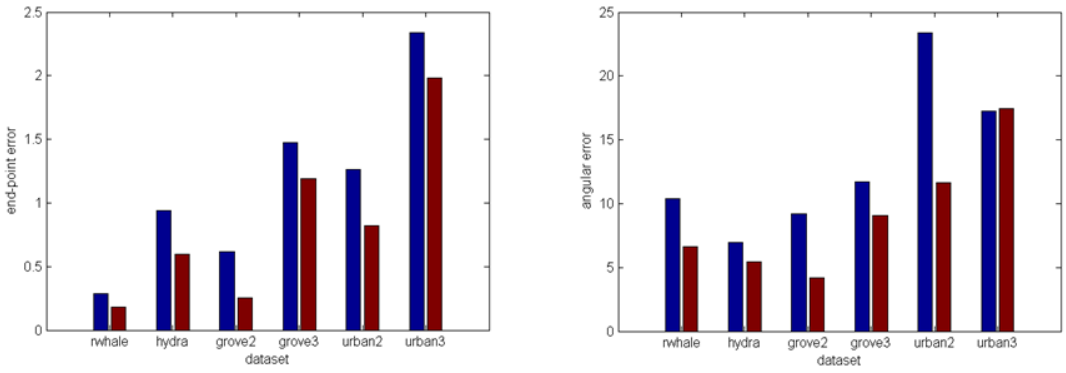

Figure 4.12: EPE and AAE comparison between the TV inpainting and spatio-temporal motion inpainting method.

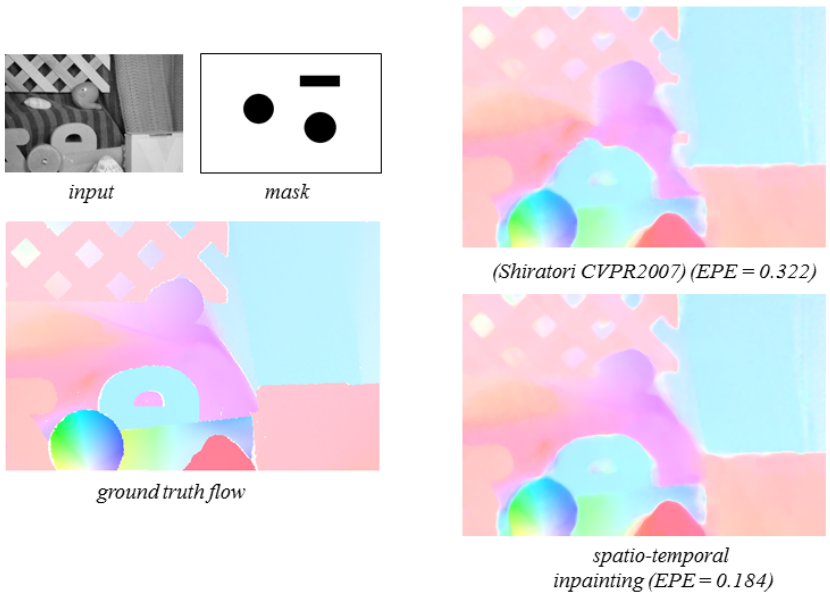

Figure 4.13: Comparison with our implementation of [Shiratori *et al.* 2006] using the rubberwhale dataset. Our method produced more accurate inpainting results.

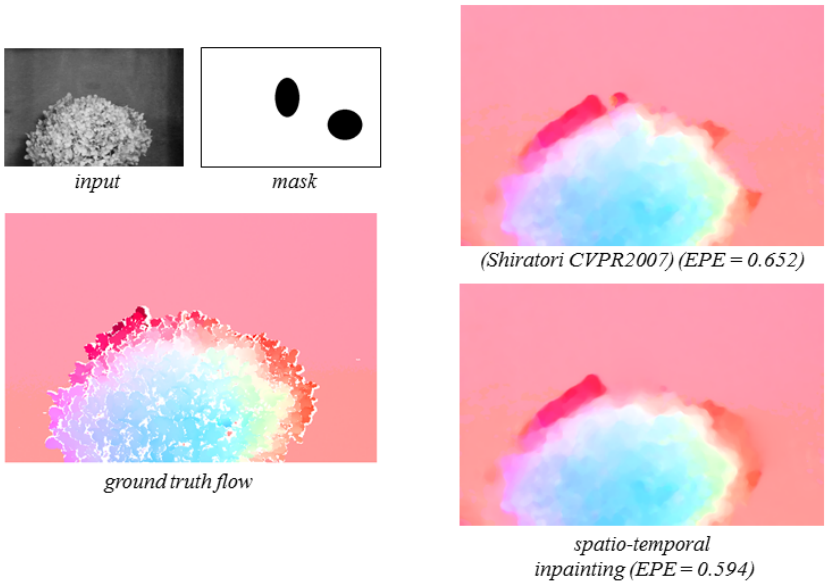

Figure 4.14: Comparison with our implementation of [Shiratori *et al.* 2006] using the hydrangea dataset. Our method produced more accurate inpainting results.

CHAPTER

FIVE

Simultaneous Motion Inpainting and Color Propagation

Contents

Using optical flow frames in video completion requires two general steps: 1) motion inpainting and 2) color propagation. We have presented in Chapter 4 how to obtain a spatio-temporally consistent motion inside the hole with the absence of color information by inferring on the optical flow of its boundaries. It is now possible to follow this motion to other frames until a known pixel is found and copy that pixel to the hole to complete the color frames of the video.

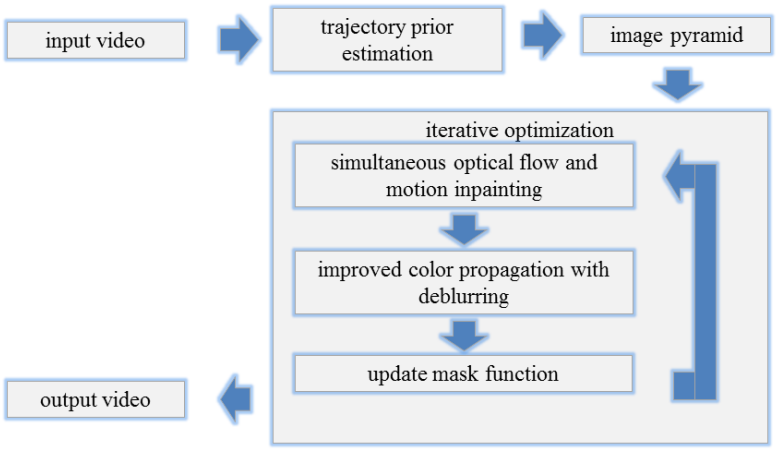

Figure 5.1: Overview of the iterative optimization technique.

However, we argue that it is still possible to use the newly inpainted color of the hole to the brightness constraint E_{data} to further improve the motion estimation. This time, instead of using a binary label for the mask of the brightness constraint, we use a probabilistic mask function that is dependent on the distance between the inpainted frame and the source frame of the color used.

With the modified mask function, we propose to combine the motion estimation and inpainting process in Chapter 4 and the color propagation method into one iterative optimization problem. The overview of the proposed method is illustrated in Figure 5.1.

In the following sections, we will discuss how to solve the trajectory constraint when the velocity is not constant. We then introduce the improved color propagation technique and then define and characterize the mask function. Then, we will show the overall method combining the motion inpainting technique in Chapter 4 and the steps that we propose here. Finally, we will show the experimental results and comparisons with existing methods followed by the chapter summary.

5.1 Trajectory Prior Estimation

The trajectory function is defined as the relationship between the forward and the backward flow. The simplest way to use the this constraint is to assume a constant velocity motion $\mathbf{u_f} + \mathbf{u_b} = 0$ and assign a small value to λ_t. However, there are several cases that this poses a problem.

The obvious case is when the velocity is not constant and the change is abrupt that the difference between two consecutive optical flow frames is large.

The other case has to do with changing perspective due to camera motion and occlusions. This problem is two-folds. First, let us assume that the motion

of the objects in the video is only due to the camera's ego motion. In the case where the camera decelerates or accelarates at a constant rate, the velocity will not be linearly varying. Although this problem can be addressed by limiting the effect of the trajectory constraint, a difficult case happens if an object being inpainted suddenly changes it's appearance due to the changing perspective.

We will limit this problem to a simple case where the motion is all due to the camera and the objects being inpainted is stationary and rigid (with respect to the world coordinate system).

Say for example we are inpainting a wall. On the other side of the hole, we the wall is seen as planar and runs at a diagonal with the camera plane. When the wall reappears on the other side of the video, it suddenly maps to a straight line. If the camera velocity is constant, it is possible to trace the motion of all the points in the wall and they will culminate as a line on the other side. The motion can then be estimated when the wall is still visible and simply follow that motion to the other side of the hole.

The problem occurs when the velocity is not constant. Take for example the second case in Figure 5.2. The wall that is being inpainted was visible on both side of the hole but with different perspective (one side is more perpendicular to the camera plane). During the time when the wall is still behind the hole, the camera is changing speeds. If we follow the same idea in the first case, we will run into a problem where tracing the motion of the points in the wall will result in a different reconstructed perspective of the wall.

Another way to view this problem is when we try to connect the motion from both sides of the hole. Take for example Figure 5.3. A constant velocity corresponds to a straight line in the epipolar plane. If we trace the movement of two corresponding points from both end of the hole, we will find that the two points will jot meet at the center.

To address the problem we have just discussed, we propose to estimate the relative motion of the points in the by reasoning on the motion of the known points. We call this relationship as the trajectory prior and can be described in the trajectory energy of the optical flow estimation as:

$$E_{trajectory} = \lambda_t \left| u_f - p_f \right|_2^2 \tag{5.1}$$

where the p_f term is the trajectory prior. The expression can be illustrated as in Figure 5.4.

We allow the forward optical flow u_f to be around the same value as the trajectory prior by minimizing their difference instead of with the backward flow. Given u_b it is possible to solve for the trajectory of u_f by using this technique. In most cases, the two directions are unknown, therefore the difference serves as a weak constraint on their estimated values.

Assuming that the motion in the video is due to the camera motion, and that the camera motion is dominantly translational with the parameters $T =$

Figure 5.2: Sequence with a wall changing its appearance on either side of the hole with the camera changing its speed at the same time.

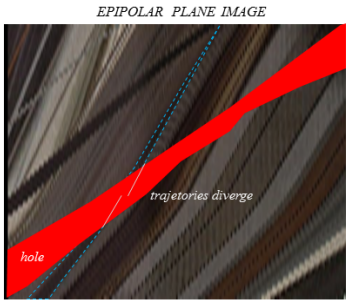

Figure 5.3: Sequence with a wall changing its appearance on either side of the hole with the camera changing its speed at the same time.

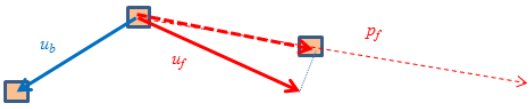

Figure 5.4: Illustration of the trajectory prior.

(T_x, T_y, T_z), the trajectory prior can be defined as:

$$p_{fx} = \frac{-T_{xf} + xT_{xf}}{-T_{xb} + xT_{xb}} u_b = \rho u_b \tag{5.2}$$

we call ρ as the transition ratio because it defines the transition of the trajectory from the backward to the forward direction. The next step of the trajectory prior estimation is to solve for this ρ from the camera motion parameters. First, we will estimate the camera parameters in the following subsection using structure from motion techniques and use that parameter to derive the transition ratio.

5.1.1 Method using Structure from Motion

One way to solve the transition ratio ρ is to derive the camera motion parameters. Assuming that this egomotion is dominantly translational, we can ignore any rotational motion and the transformation matrix can be written as:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{5.3}$$

We first solve the point correspondence using scale invariant feature transform (SIFT) [Lowe 2004]. Then using the matching points, we solve for the camera translation [Hartley & Zisserman 2004]. We then use the camera translation parameters to derive the transition ratio. We test this approach on a synthetic video where the camera is moving along the x-axis in constant velocity and suddenly changes its speed (see Figure 5.5. The derived translation in the x-axis (T_x is shown in the bottom left plot and the derived transition ration on the bottom right plot. We can observe that the transition ratio has an abrupt dip in the value at around the tenth frame and goes back to its almost constant range.

There are several issues in using this method in solving for the transition ratio of all the frames. First, inherent to structure from motion techniques is the requirement of having several features to be visible on all frames being used. Apparently, this is hardly the case, especially when the camera is moving fast. Also, for the estimation to be accurate, the method requires many frames and it takes a long time to be estimated.

5.1.2 Method using Point Correspondences

We propose another method in solving for the trajectory prior. Instead of solving for the camera parameters, we utilize the definition of the transition ratio, where in we could solve the ratio between the camera motion if we assume that either of the motion is constant. For example, if we assume that the camera does not

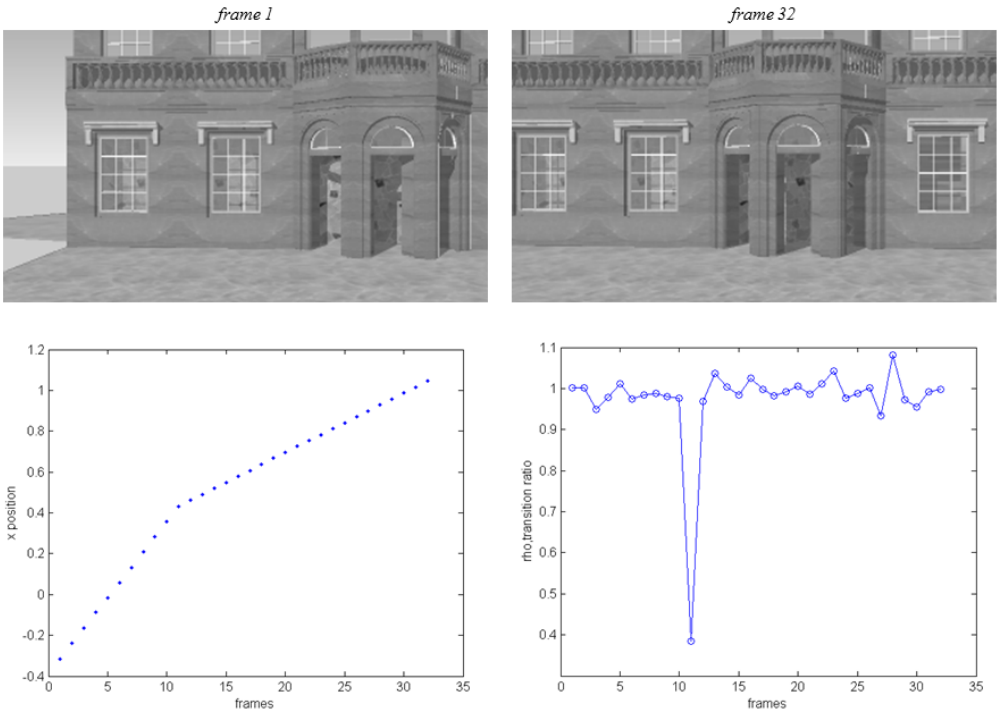

Figure 5.5: Top row: input frames. Bottom row: (left) camera translation along the x-axis, which shows a sudden change in speed about the 11th frame, and (right) calculated transition ratio. Notice the spike at the 11th frame which indicates a sudden change in speed.

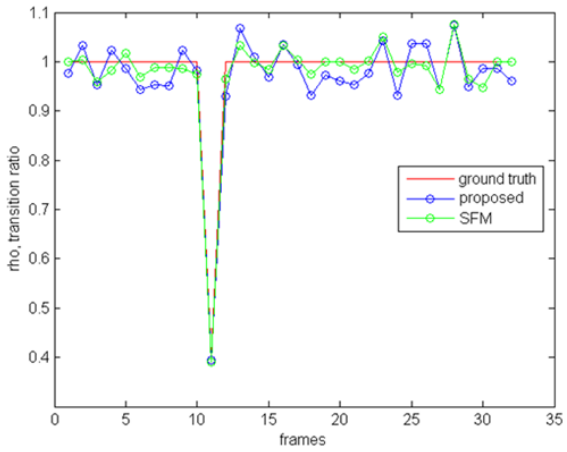

Figure 5.6: Comparison in the calculated transition ratio between the methods using SFM and only point correspondences.

change its depth much compared to its motion along the x-axis, we can ignore the contribution of the T_z parameter in the equation. We then approximate the transition ratio as:

$$\rho \approx \frac{T_{xf}}{T_{xb}} \tag{5.4}$$

We can solve ρ instead by taking the average ratio of the known optical flow u_{fi} and u_{bi} of all i outside the hole. We first solve again the SIFT features, this time only between three frames. The transition ratio is then given by:

$$\rho = \frac{1}{N} \sum_{i}^{N} \frac{u_{fi}}{u_{bi}} \tag{5.5}$$

We compare the result of this method with the one using structure from motion in Figure 5.6. We show that this method is very close to the result of the SFM with several advantages. First, it only requires three frames to be able to estimate all the average ratio. Moreover, with the reduction in frames, we can estimate ρ very fast. Moreover, since we are already given the optical flows of the known points outside the hole, this step can be calculated only once.

5.2 Prior Color Propagation Techniques

We first present a simple color propagation technique used in [Shiratori *et al.* 2006] based on linear warping and pose the problems that it implies.

The method starts with an inpainted motion inside the hole. The values of the optical flow in each frames forms a graph that maps the pixels between

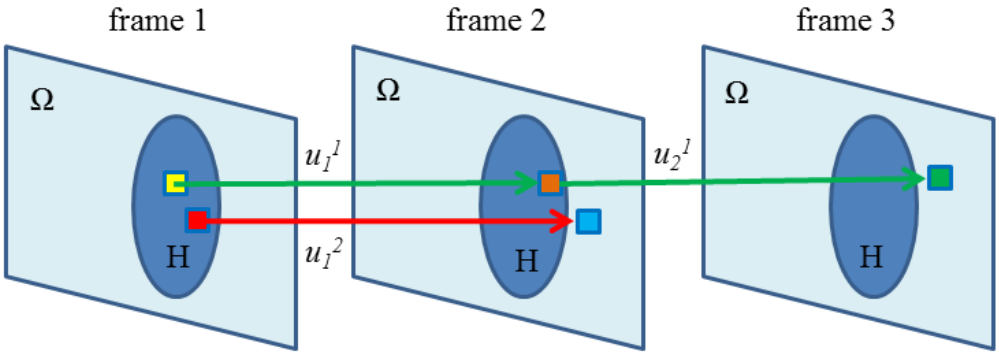

Figure 5.7: Color propagation as graph. The optical flow is treated as undirected graphs that maps the correspondence of pixels among the frames.

neighboring frames. The optical flow is treated as an undirected edges and is illustrated in Figure 5.7.

By following a graph technique, the color propagation method can handle several situations. Take for example the red pixel in frame 1. Its optical flow u_1^2 directs to a known pixel in frame 2 represented by the blue pixel. This blue pixel will then be copied to the position of the red pixel, thus completing the color.

Practically speaking, only the values close to the boundary of the hole will have a mapping on the known pixel in the succeeding frame. Unfortunately, the holes closer to the center points, in most cases, points to another hole in the next frame. Take for example the yellow pixel in frame 1. By following its optical flow u_1^1, we can see that it directs to an unknown pixel in frame 2 labeled by the orange pixel. Obviously, we cannot copy this pixel to frame 1, thus we need to find another mapping for the yellow one. If we again trace the flow of the orange pixel, we can see that it points now to a known pixel in frame 3 labeled by the green pixel. We then copy this green pixel to the orange position thus completing the hole in frame 2. In another run of the color propagation, the yellow pixel in frame 1 will now point to a known orange pixel, thus we copy this value to the position of the yellow one and completing the hole.

Ideally, we want the optical flows to point to an exact location in another frame. However, this is seldom the case. A motion vector usually always point to an inexact pixel and therefore the mapping of the pixels become incomplete.

A quick solution to this problem is to discretize the values of the motion vector using nearest neighbor and winner-take-all algorithm. To do this, a pixel closest to where the motion points become the color in the hole. This solution, however, results in cases where a multiple flow points to a single pixel and therefore allows spatial discontinuity (or gaps) in the inpainted hole.

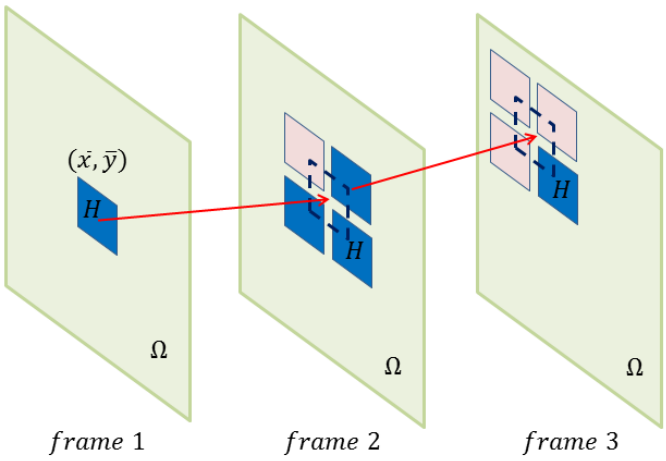

Figure 5.8: Color propagation as graph. The optical flow is treated as undirected graphs that maps the correspondence of pixels among the frames.

Consider the case in Figure 5.8. To inpaint the hole in frame 1, [Shiratori *et al.* 2006] proposed to use the sizes of the overlapping areas in the intermediate frame 2 as weights to determine the contribution of each of the pixels. The color of the hole in frame 1 is then a weighted average of the four neighboring pixels. Assuming that the weight is given by $w(x, y)$, the color $c(\bar{x}, \bar{y})$ is given by:

$$c(\bar{x}, \bar{y}) = \frac{\sum_{x,y} w(x, y)c(x, y)}{\sum_{x,y} w(x, y)} \tag{5.6}$$

Another way to think of this problem is to warp the known pixels to the hole via bicubic interpolation (see Chapter 4). The four pixels in the vicinity of the hole and their neighboring pixels is used as the initial values of the interpolation method. The image derivatives are solved about these four points and the value of $(\bar{x}, \bar{y})$ is determined using a bicubic polynomial.

Although the color propagation technique described above will work, several issues still need to be addressed. First for every pixel inside the hole, we have estimated optical flow in two directions, namely backward and forward flow. Problem arises when these two pixels point to different values. Secondly, propagating a color from a distant frame using linear warping causes the inpainted color to be blurred. This happens when the flow points to another hole in the next frame. Since the color in that position is already a warped value, warping it again to the current frame causes the color to be further smoothed.

5.3 Improved Color Propagation Method

5.3.1 Handling Two Directions (Forward and Backward Flow)

We perform the color propagation technique in Section 5.2 for each of the forward and backward flow directions. As an additional step, we record/save the distance between the current frame and the source frame. Say, we are completing the n^{th} frame I_n (current frame) of the image sequence. For each pixel x_i, we are propagating the color $\bar{x}_i$ of the known region in $(n+s_i)^{th}$ frame I_{n+s_i}. We define a distance $\mu(x)$ as:

$$\mu(x) = (n + s_i) - (n) = s_i. \tag{5.7}$$

After color propagation, we will have two differently inpainted frames I_{H_b} and I_{H_f} (see Figure 5.9). If we calculate the first order derivative of each frame, we can see a large discontinuity in the spatial smoothness of the inpainted color around the boundaries of the hole. If we look closely, a discontinuity occurs on the leading edge (labeled by the yellow dashed line in the second row) of the hole in either directions.

We show in the third column of Figure 5.9 the map of the difference of the image gradients of the leading edge of the forward flow and the lagging edge of the backward flow and vice versa. As we can see, the gradients is very steep along the boundary due to 1) discontinuity in the color of the hole and the boundary and 2) difference in the smoothness at different edges. In the last column, we show the sampled value at the middle row pixel of the gradient map for clearer visualization of the smoothness gap.

The error along the leading edge boundary is correlated $\mu(x)$. It appears that as the distance of the source frame increases, the discontinuity in spatial smoothness becomes more apparent. In Figure 5.10, we show the value of the $\mu(x)$ with the white value being highest and black as the lowest. Obviously, outside the hole, this value is zero. Inside the hole, the leading edges of both directions have a larger μ and gradually decreases as we go to the lagging edge. Simply saying, the discontinuity in the smoothness of the inpainted hole increases as the source frame of that hole moves farther away from the reference frame.

We can also interpret this in another way. Take for instance the forward direction frame. Its leading edge correspond to the direction which the object moves in the opposite direction of the camera. In other words, as the hole moves, it occludes more object and hides the background more and more towards the center of the hole. What happens then is that it becomes almost impossible for us to know the value at the leading edge in the forward direction if we continue on moving in this direction.

On the other hand, the lagging edge in the forward direction reveals more background as it moves along. Hence, we can interpret the value at the lagging

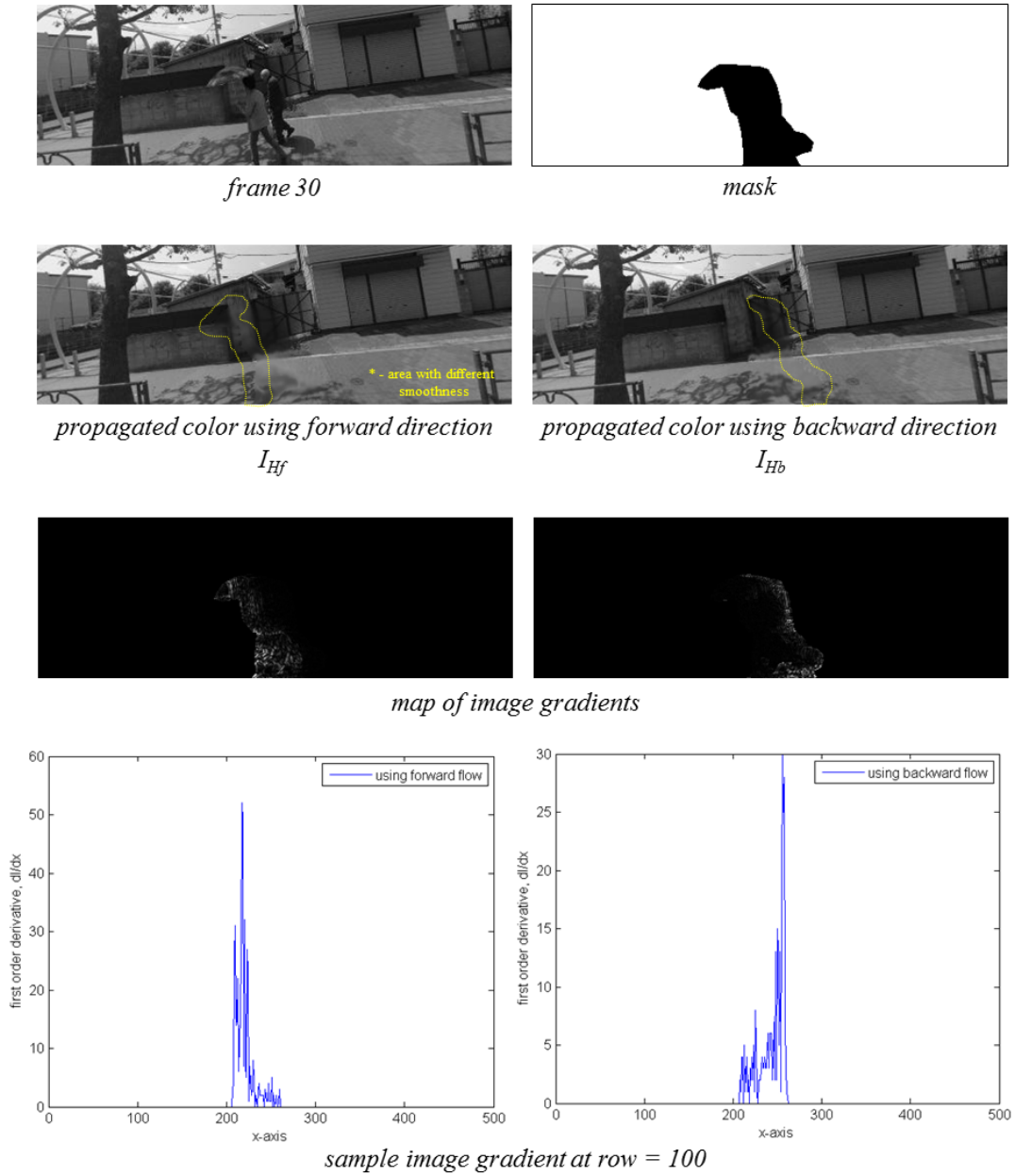

Figure 5.9: Forward and backward image, show the leading edge and lagging edge, show gradient, show line plot for one line

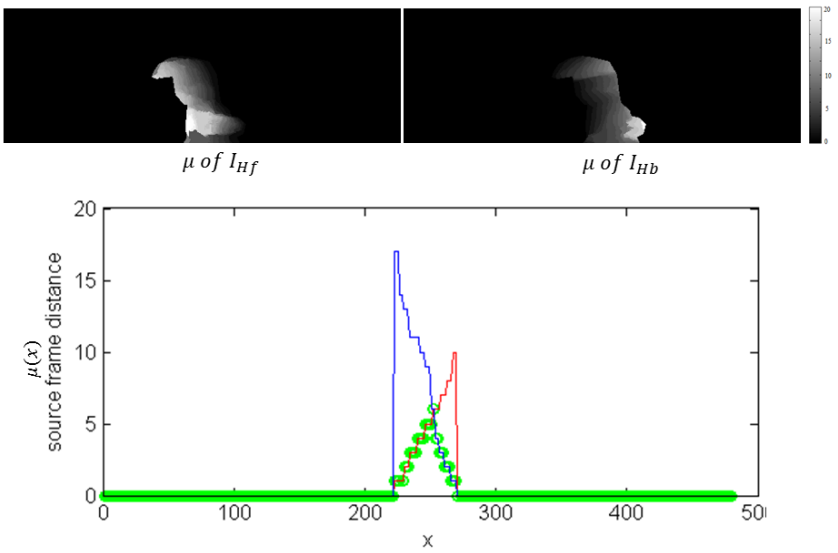

Figure 5.10: $\mu(x)$

edge almost immediately as the hole uncovers it. Obviously, at the lagging edge, it becomes easier to fill up the hole.

It is important to note though, that at the lagging edge of the hole, the inpainted color is very accurate. And since we have two lagging edges that compensate each other, we can use their differences to our advantage. This means that we can combine the results of both directions correctly and reduce the discontinuity error at the leading edges.

We perform this combination using two steps. First, in regions close to the hole boundary, we choose the direction which has the lower μ value. As we move further inside the hole, we blend the color from both directions using:

$$I_H = \frac{\mu_b^2}{\mu_b^2 + \mu_f^2} I_{H_f} + \frac{\mu_f^2}{\mu_b^2 + \mu_f^2} I_{H_b} \tag{5.8}$$

We show the comparison in Figure 5.11 between the inpainted result using both direction with and without using the blending technique in Equation 5.8. Looking at the original frame for the reference of where the deleted part is (this case the pedestrians are removed, thus corresponding to the position of the hole), we can see that without blending, there is a sharp edge in somewhere in the middle of the hole. Fortunately, the part along the boundary of the hole seems to be smooth and without detectable breaks.

We improved the output using the blending technique around the center of the hole. In the bottom image of Figure 5.11, we were able to remove the sharp edge around the center of the hole. This improvement is two-folds. First, the

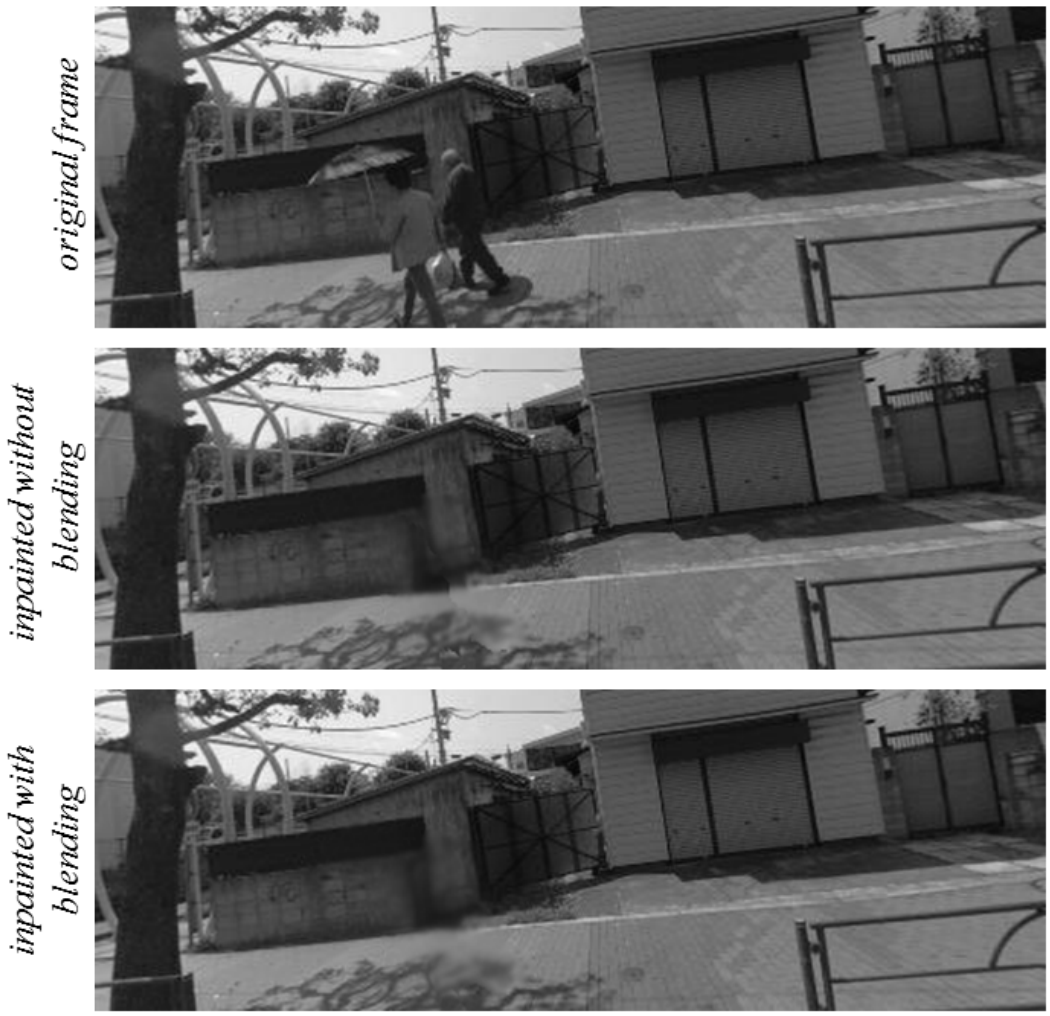

Figure 5.11: Comparison between blended and non-blended directions

blending technique obviously smooths the results between the forward and the backward direction. By increasing the area being blended, we can make are smoother. However, this will result in a smoothness that is apparent to the viewer.

Fortunately, this is not very case. As we will discuss in the succeeding chapters, the improvement is propagated from the coarsest scale of the image pyramid. Therefore, as we improved the result in the coarsest scale, the smoothness gradually reduces because we are able to compute a more accurate color in the hole. If the color between the propagated values using the forward and the backward direction is very close, the smoothing effect is greatly reduced.

5.3.2 Reducing the Blurring Effect Due to Warping

Effect of Consecutive Warping

We also address the blurring effect caused by the linear warping technique and increasing μ. As we have observed in our experiments, consecutive linear warping results in a blurred inpainted color. Since the color of the pixel in the hole is only an interpolated value of the surrounding pixels, subsequent interpolation will result in averaging effect. This effect is more apparent with large holes which inner parts can only be inpainted using sources from distant frames.

To be more clear, we illustrate this problem with a synthetic video. In Figure 5.12, we introduce a hole with increasing depth (number of frames) and show the qualitative change in the blurring.

As we can see, as the blurring seem more apparent as the hole depth increases. We also compute the interpolation error (the difference between the inpainted frame and the ground truth frame) and plot (in Figure 5.13) the result with respect to increasing hole depth. The interpolation error (IE) [Baker *et al.* 2007] is computed as in Equation 5.9.

$$IE = \sqrt{\frac{1}{N}\sum_{(x,y)} \left(I(x,y) - I_{GT}(x,y)\right)^2} \tag{5.9}$$

Reducing the Blurring Effect

To reduce the blurring effect, we use the following techniques illustrated in Figure 5.14. As of now, we are inpainting a frame by interpolating the values in the source frame on a frame-by-frame basis. Say for example, we are completing the hole in frame 1. We do this by propagating the color following $u_{(1,2)}$ to frame 2 and if this points to the hole, we follow to $u_{2,3}$ and so on. Then, we warp the colors from frame 3 to frame 2 using $u_{(2,3)}$ and then the colors from frame 2 to frame 1 using $u(1,2)$.

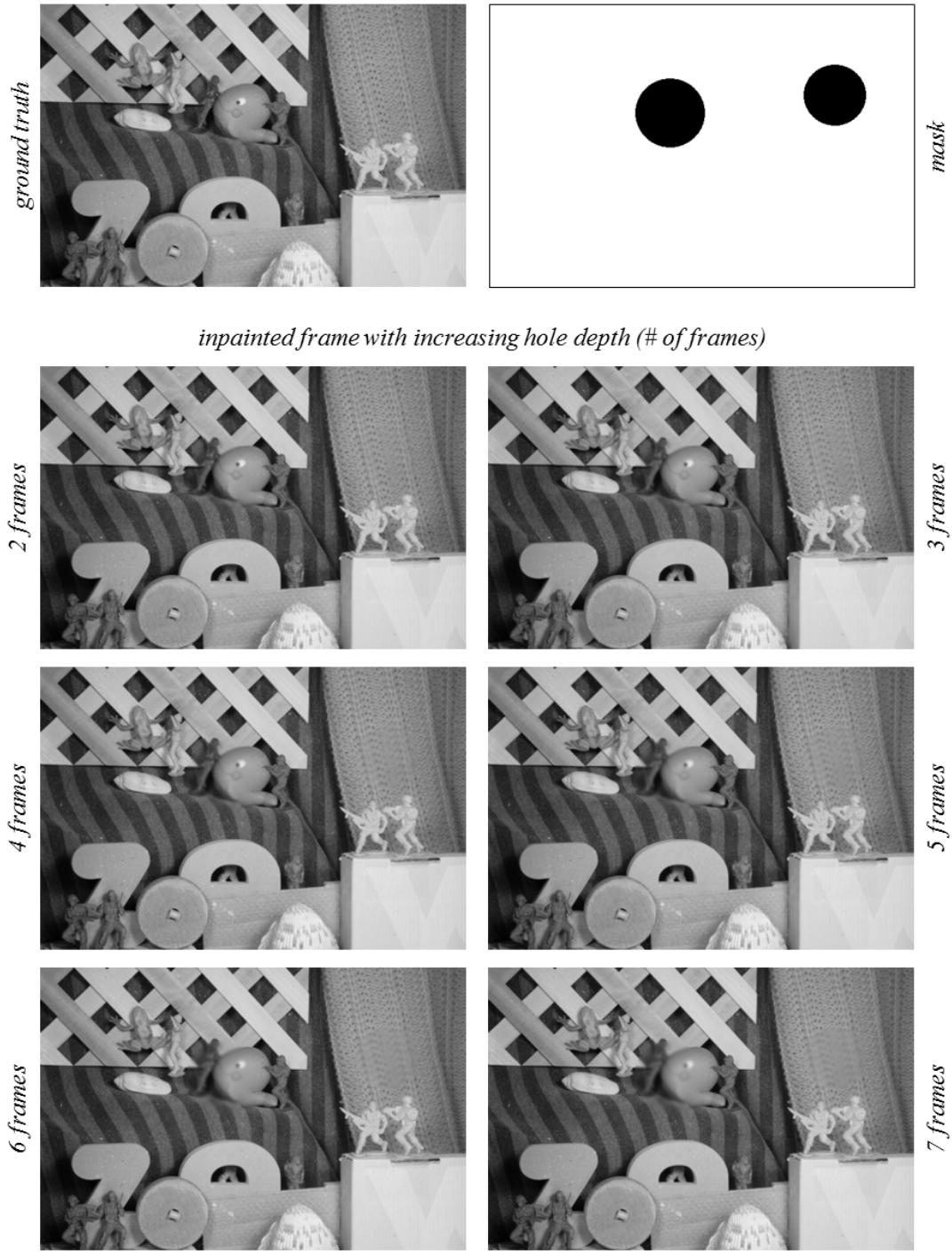

Figure 5.12: Blurring effect as the source frame distance increases. The hole is increased in increments to illustrate the blurring effect and frame distance relationship.

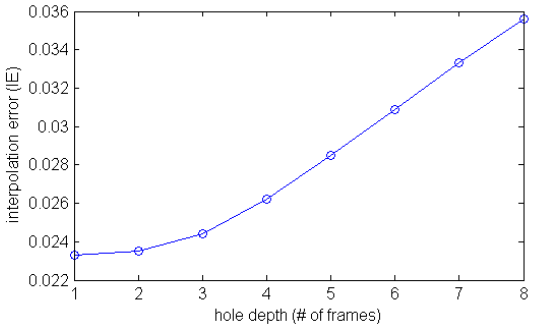

Figure 5.13: Interpolation error due to the blurring effect with increasing hole depth. The error increases as the hole becomes deeper which means that the source frame of the hole moves farther away from the reference frame and therefore incurs more interpolated result.

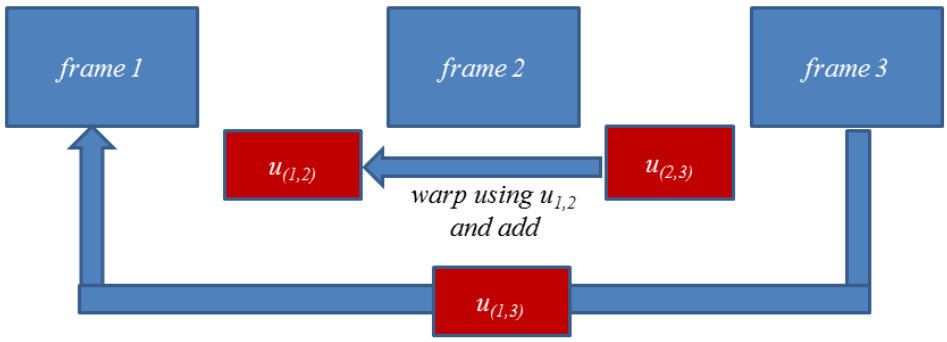

Figure 5.14: Warping of motion.

We propose that instead of the above method, we directly interpolate the values in frame 3 to frame 1 by solving for the optical flow $u_{(1,3)}$. Fortunately, we do not need to solve for this again.

First, we solve for $\mu(x)$ of all the frames using the method in Section 5.3.1. The $\mu(x)$ contains the frame location of the source pixel in all x inside the hole. To illustrate, say, we are completing the n^{th} frame I_n. For a pixel x in the hole in I_n, we have the location of the source frame $n + \mu(x)$. We then linearly warp the optical flow of that frame $u_{n+\mu(x)}$ using the flow of the proceeding frame $u_{n+\mu(x)-1}$. We iterate this linear warping kk times until we get to the current frame $n + \mu(x) - k = n$. As a result, we will get the optical flow $u_{n+\mu(x)}$ of point x that maps it to frame $n + \mu(x)$. We repeat this process for all the pixels in parallel.

With the updated motion, we update I_{H_b} and I_{H_f} and perform the weighted combination to get I_H as in Equation 5.8.

5.4 Mask Function

So far, we are able to create a completed video through our proposed motion inpainting and color propagation methods. However, we argue that the color in the initially completed hole can still be used to further improve the result of the motion inpainting.

To do this, we modify the binary label mask function $m(x)$ to have a spatially varying value based on the reliability of the inpainted pixel. We define the reliability of the pixel as:

$$m(x) = \gamma^{-mu(x)} \tag{5.10}$$

where γ is a positive real number. The value of gamma controls how much the inpainted pixel affects the overall error.

Choosing an arbitrary γ value will result in unstable minimization. In theory, we want the total error inside the whole to be less than that of its boundary [Wexler & Irani 2007] [Criminisi *et al.* 2003]. This will help in the convergence and allows the information to gradually propagate towards the hole. Choosing a small value of γ however, will let the newly inpainted color at the center of the hole more effect on the minimization rather than the spatial and trajectory smoothness. This results in wrong inpainted motion. On the other hand, a very large value results in the information not reaching the center of the hole, especially if it is too big. In our experiments, we choose $\gamma = 1.3$ and find this value suitable in most situations.

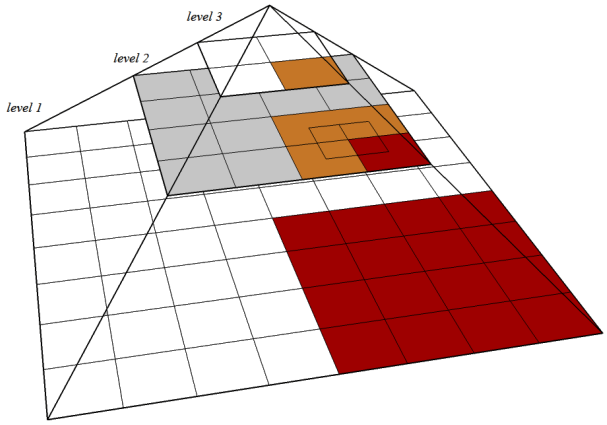

Figure 5.15: Image Pyramids

5.5 Iterative Inpainting and Color Propagation Method

The first stage of the iterative motion inpainting and color propagation is to solve for the image pyramids (see Figure 5.15) through a coarse to fine strategy [Burt 1981], [Burt 1983], [Burt & Adelson 1983]. We do this by repeatedly down-sampling the image by a factor of α. The higher level $l+1$ image (or the coarser scale) given the current level l image G_l is solve as:

$$G_{l+1}(x,y) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} 0.25 G_l(2x+m, 2y+n) \tag{5.11}$$

Using this approach, we compensate for large pixel motions that is usually present in our videos. We use $\alpha > 0.5$ so that each of the succeeding pyramid is a blurred version of the lower pyramid. For this implementation we limit the size of the pyramid by allowing the width of the coarsest scale image to be greater than 30 pixels.

We start the iteration from the coarsest level of the pyramid. We use an initial value of the mask to be zero inside the hole. We then orderly choose a frame, I_0, and its two neighboring frames I_f and I_b. The unknowns (u_b, u_f, b^k) are initialized to zero. We then iterate the joint motion estimation and inpainting technique described in Chapter 4, the color propagation method and the mask update discussed in this chapter.

The result after the motion inpainting are the optical flows u_f and u_b of all the frames. We perform one sweep of the inpainting process and use the output to do the color propagation. We then get from here the value for $\mu(x)$ which we use to update the mask function.

We repeat this process in the same level of the pyramid until we reach convergence. We define the convergence as a lower limit in the mean difference between the colors of successive inpainted frames for every iteration. We summarize the steps of this method in Algorithm 3.

Algorithm 3: Iterative motion inpainting and color propagation.

Require: color of H
  solve $trajectory_prior$
  solve $image\_pyramids$
  initialize $m(x \in H) = 0$
  for $level < max\_level$ **do**
    while $error > thresh$ **do**
      Inner Iteration
      Color Propagation
      update $m(x)$
    end while
    upsample u_f, u_b
  end for

After we reach the threshold, we up-sample the optical flows by $\frac{1}{\alpha}$ again by using *bicubic interpolation* and perform the iterative step again until we get to the finest level of the pyramid.

5.6 Experimental Results

5.6.1 Videos with Changing Velocities

We first test the effectiveness of the trajectory prior estimation method with different videos. We first used a video where the camera suddenly change its velocity. We compare the results between the trajectory prior estimation using SFM and point correspondence. We also compare them from the result of a constant velocity assumption. We show the representative frames in Figure **??** and plot the error (difference between the ground truth video and the inpainted video) in Figure 5.18.

We also use a shaking video (in x-axis only) to demonstrate the effectiveness of point correspondence method in solving the trajectory prior. We show an improvement in the inpainting result and show them in Figures 5.17 and 5.18.

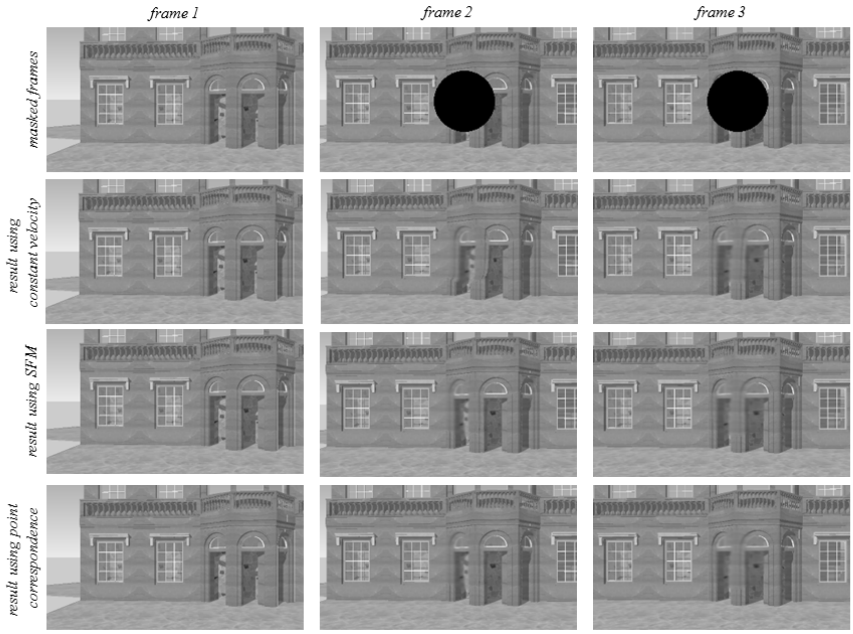

Figure 5.16: Representative frames of the result of the video completion method on a synthetic video with changing velocity. We compare the results using a constant velocity trajectory assumption and with the trajectory prior solutions using SFM and point correspondences.

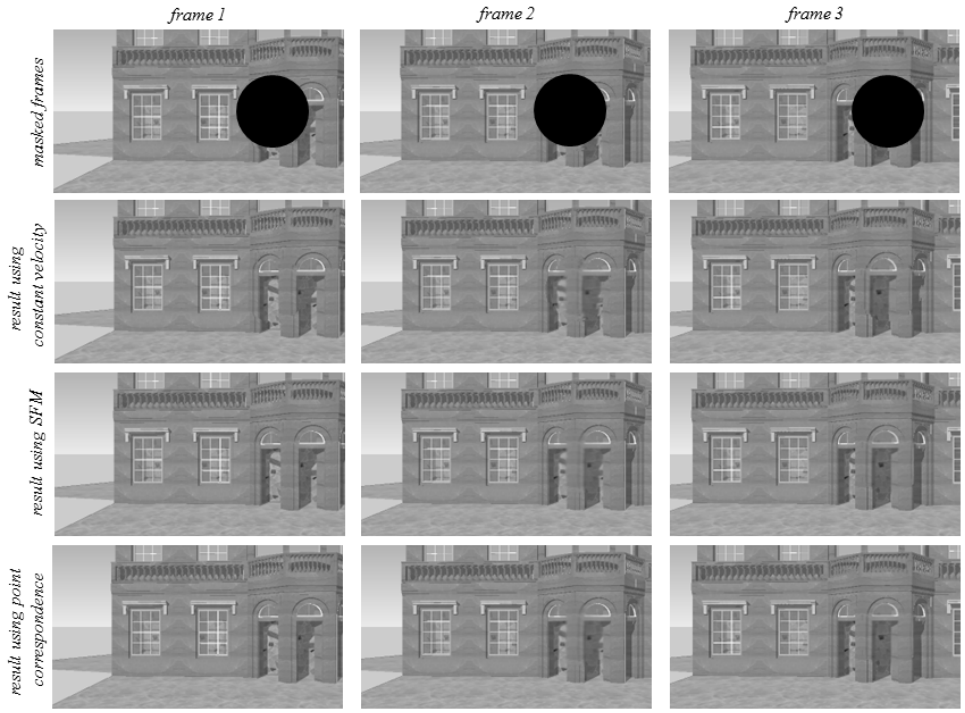

Figure 5.17: Representative frames of the result of the video completion method on a synthetic video with shaking. We compare the results using a constant velocity trajectory assumption and with the trajectory prior solutions using SFM and point correspondences.

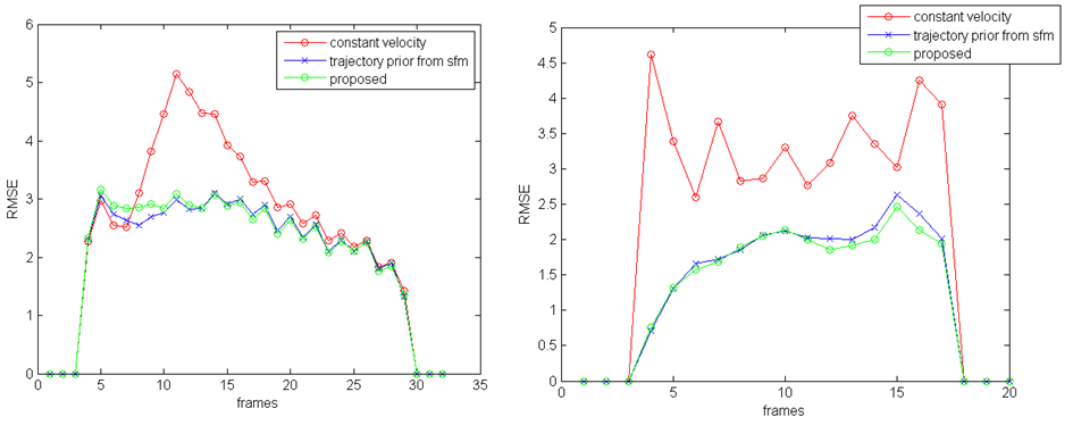

Figure 5.18: Plot of the error in completion of video with (left) changing velocity and (right) shaking. In both cases, the trajectory prior solution shows a significant reduction in error.

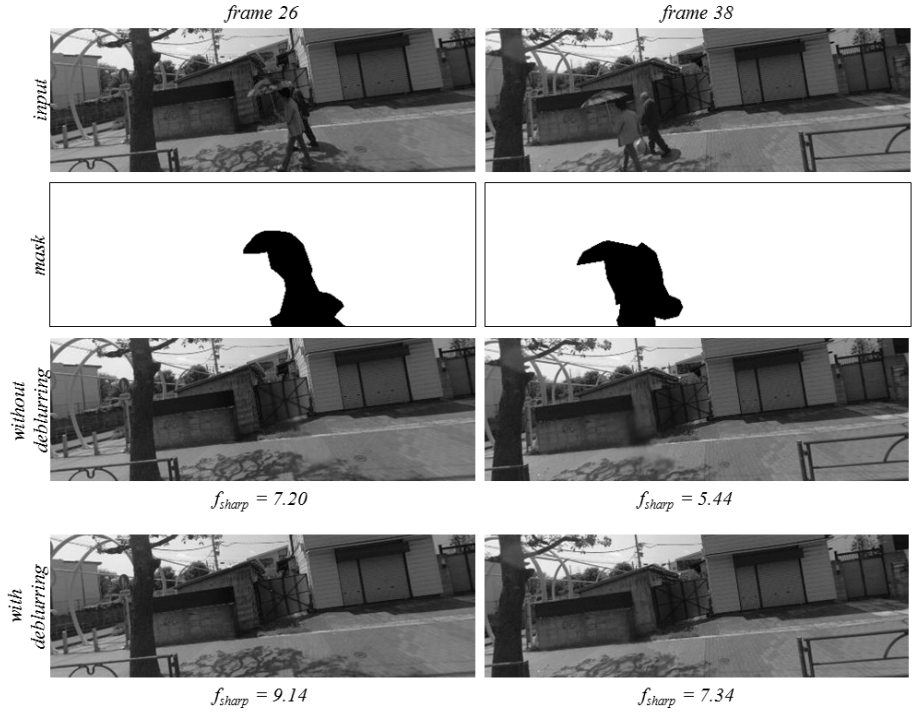

Figure 5.19: Result of reducing the blurring effect on the 'humanc' sequence for representative frames 26 and 38.

5.6.2 Blur Reduction Tests

We quantify the improvement in the results by measuring the gradient-based sharpness measure (Tenengrad function [Krotkov 1989]:

$$f_{sharp}(I) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\left(\frac{dI_i}{dx}\right)^2 + \left(\frac{dI_i}{dy}\right)^2} \tag{5.12}$$

We tested the technique using several image sequences and show the improvement in Figure 5.19. We also show the comparison between the sharpness measure of the ground truth sequence and the inpainted ones in Figure 5.20 and 5.21.

5.6.3 Effect of Mask Function

We compare the result of the motion inpainting using the new mask function and the binary label and calculate the end-point error with the ground truth. We show the results in Figures 5.22 - 5.25.

Using this method, we consistently improved the sharpness of the resulting inpainted image. However, it is important to note that the remaining blurring effect now only comes from the one-time linear warping and the blending of

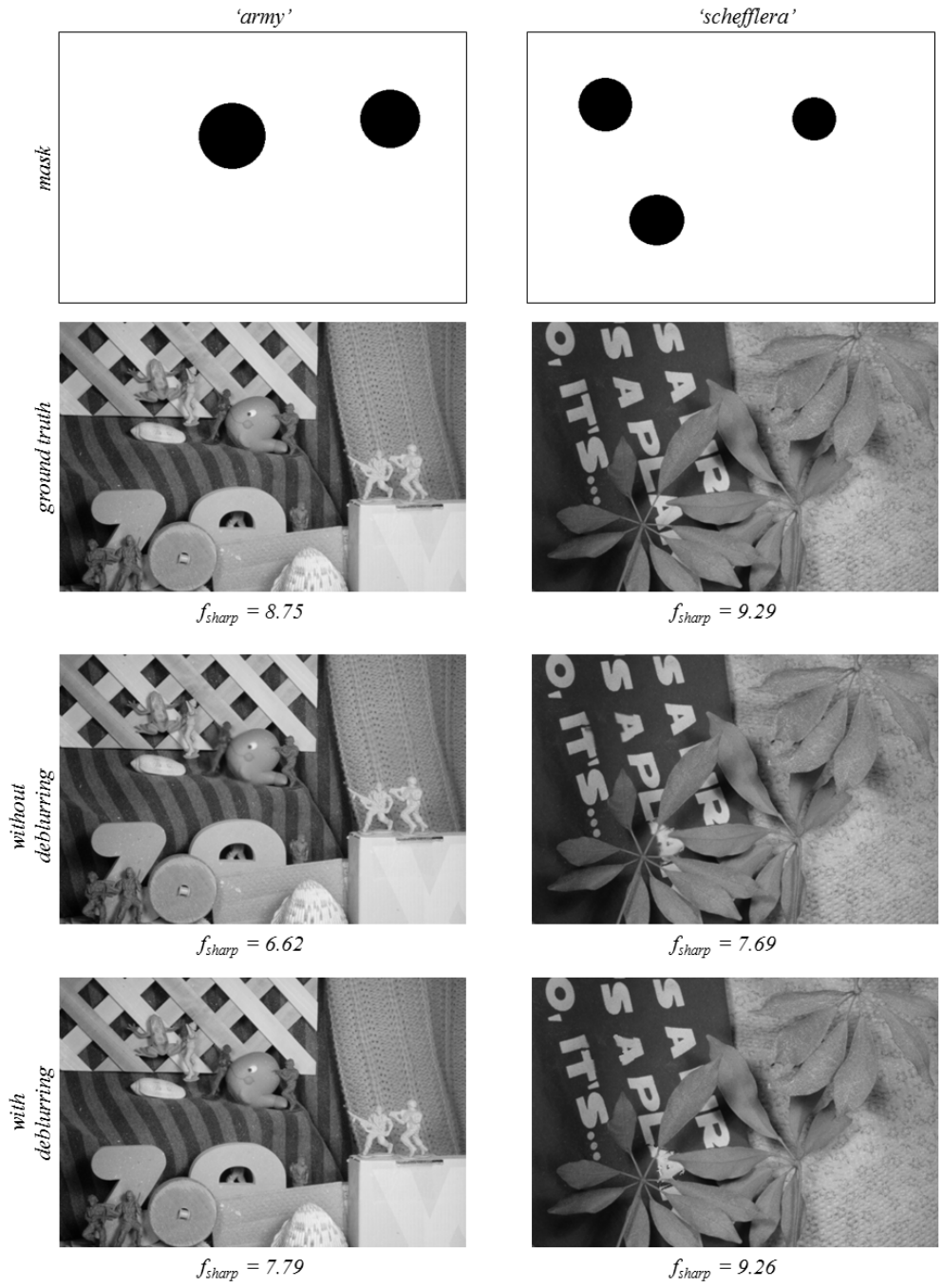

Figure 5.20: Result of reducing the blurring effect on the 'army' and 'schefflera' sequence.

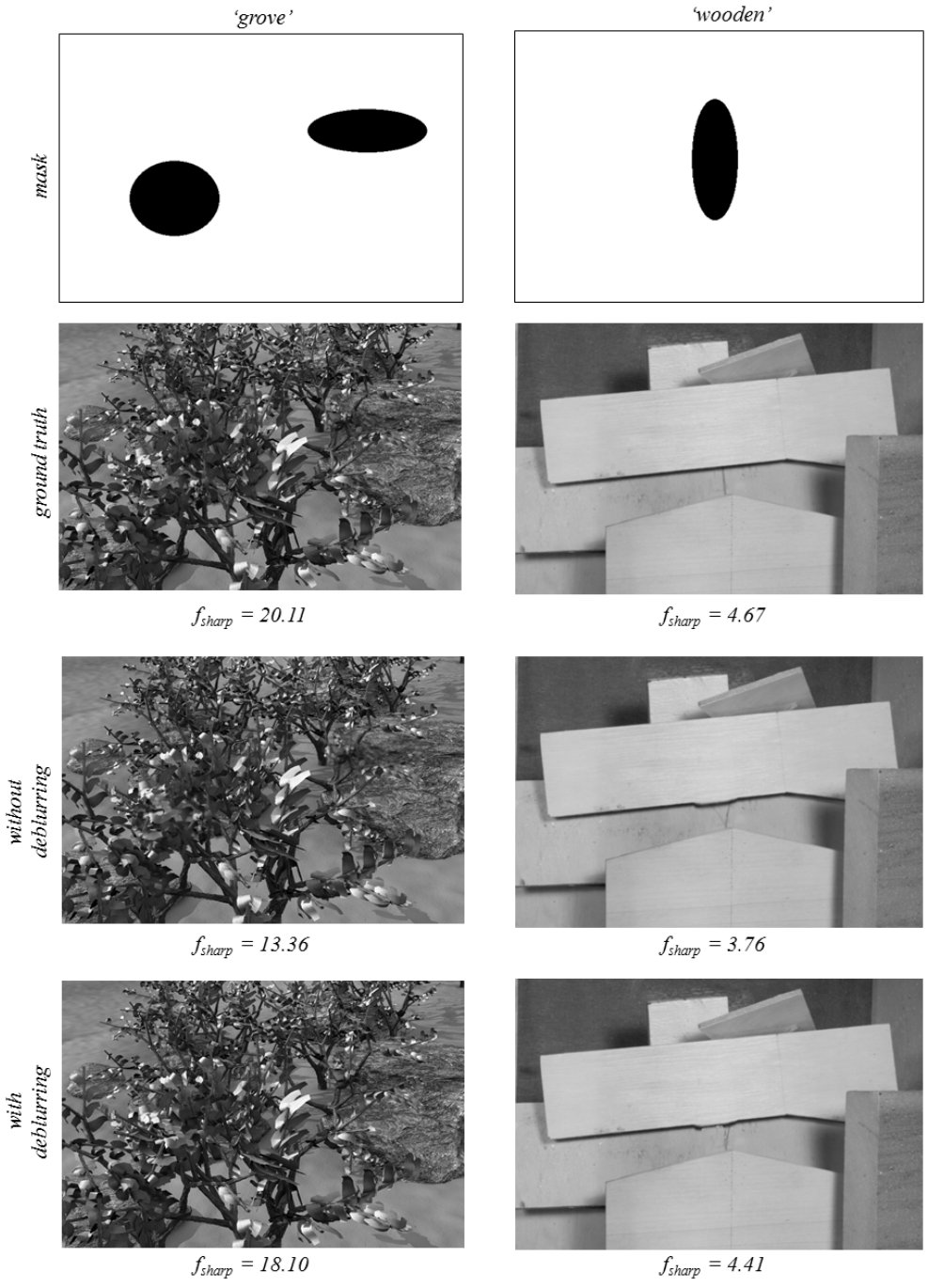

Figure 5.21: Result of reducing the blurring effect on the 'grove' and 'wooden' sequence.

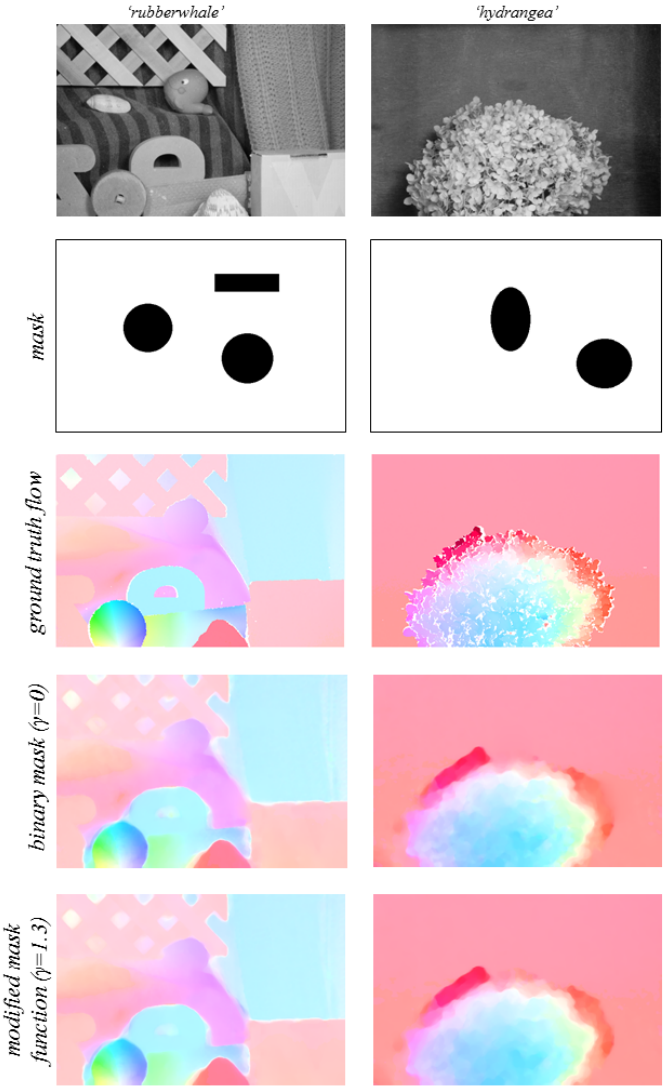

Figure 5.22: Inpainted motion with $\gamma = 1.3$ vs. $\gamma = 0$.

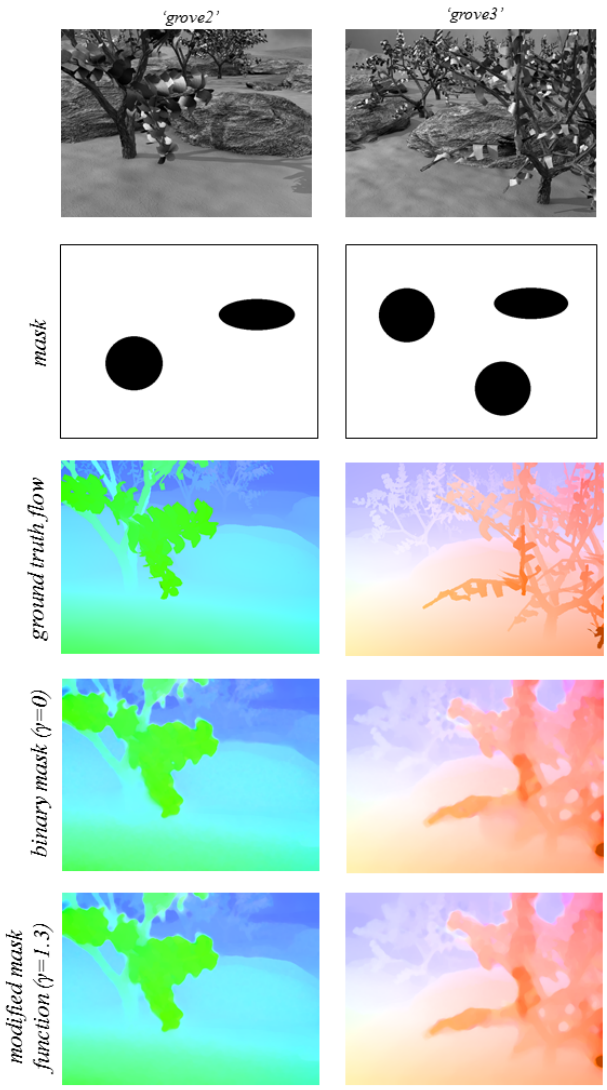

Figure 5.23: Inpainted motion with $\gamma = 1.3$ vs. $\gamma = 0$.

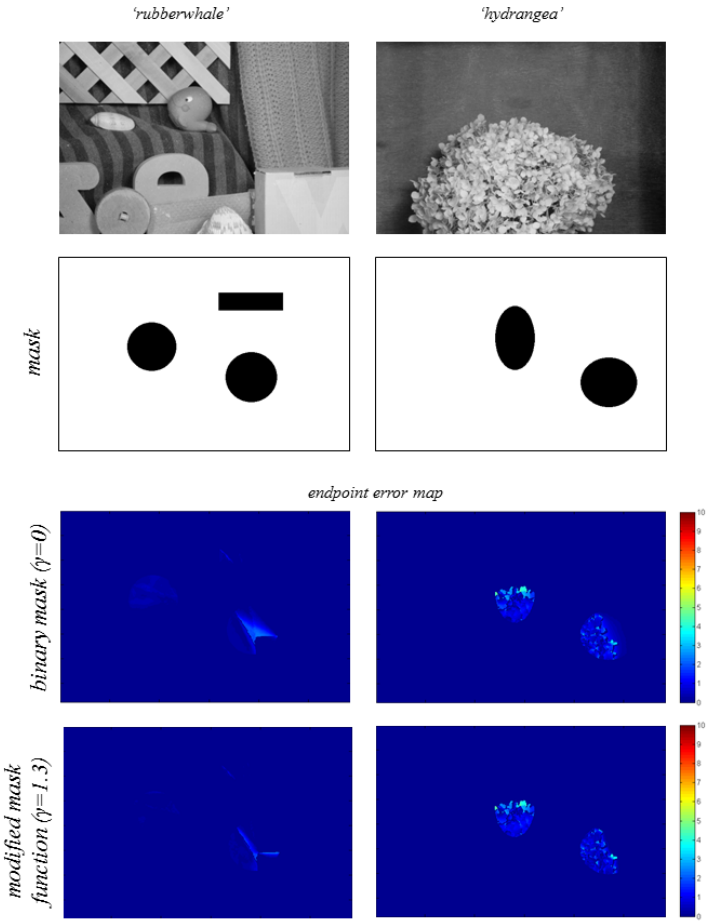

Figure 5.24: Endpoint error map of the inpainted motion with $\gamma = 1.3$ vs. $\gamma = 0$.

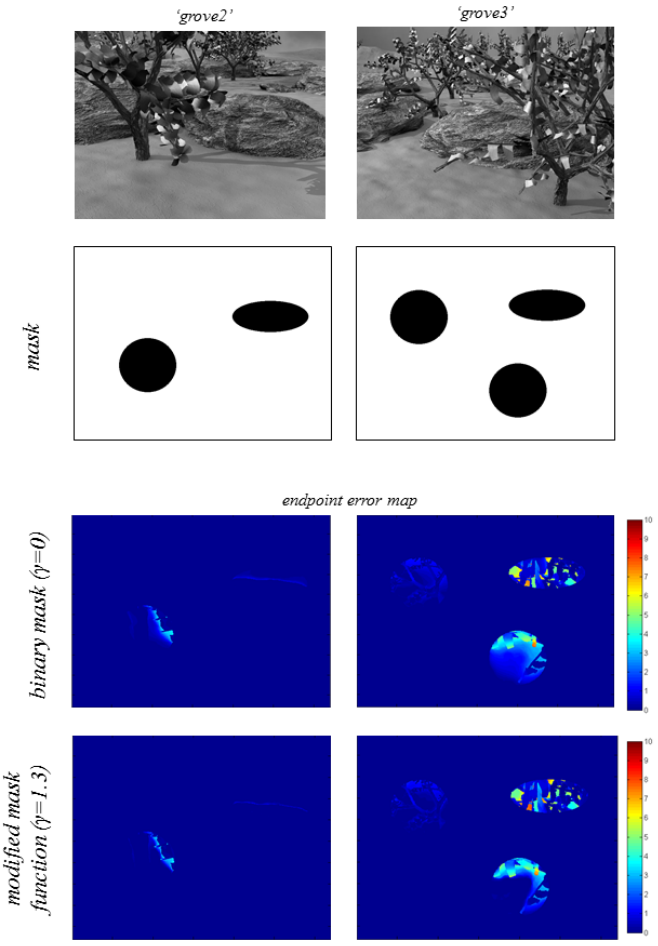

Figure 5.25: Endpoint error map of the inpainted motion with $\gamma = 1.3$ vs. $\gamma = 0$.

forward and backward propagation around the center of the hole.

5.6.4 Test on Street Videos

We tested the whole video completion process on real street videos where we remove the walking pedestrians. We show the representative frames of two image sequences 'human' and 'person' in Figures 5.26 and 5.27.

5.7 Chapter Summary

In this chapter, we proposed a simultaneous motion inpainting and color propagation method by using an iterative optimization method. We obtained better results when we used the newly inpainted pixels inside the hole to refine the optical flow estimation inside it. We control the effect of the newly inpainted pixels using our proposed mask function that relates the frame distance of the source pixel to the reference pixel in the hole. We also introduced a trajectory prior estimation method to handle the trajectory constraint during non-smooth motion. Our method comprised of only three frames and therefore was implemented really fast. We also improved the standard color propagation method to include a technique in combining the result of two directions, namely the forward and the backwards. We combined the propagated color from both directions using our proposed blending technique. We then showed in our result that this method can accomplish video completion results accurately.

One issue that was not addressed in this chapter is that the inpainted motion does not have refined details and the boundaries of motion are not well-defined. We address this issue in the next chapter.

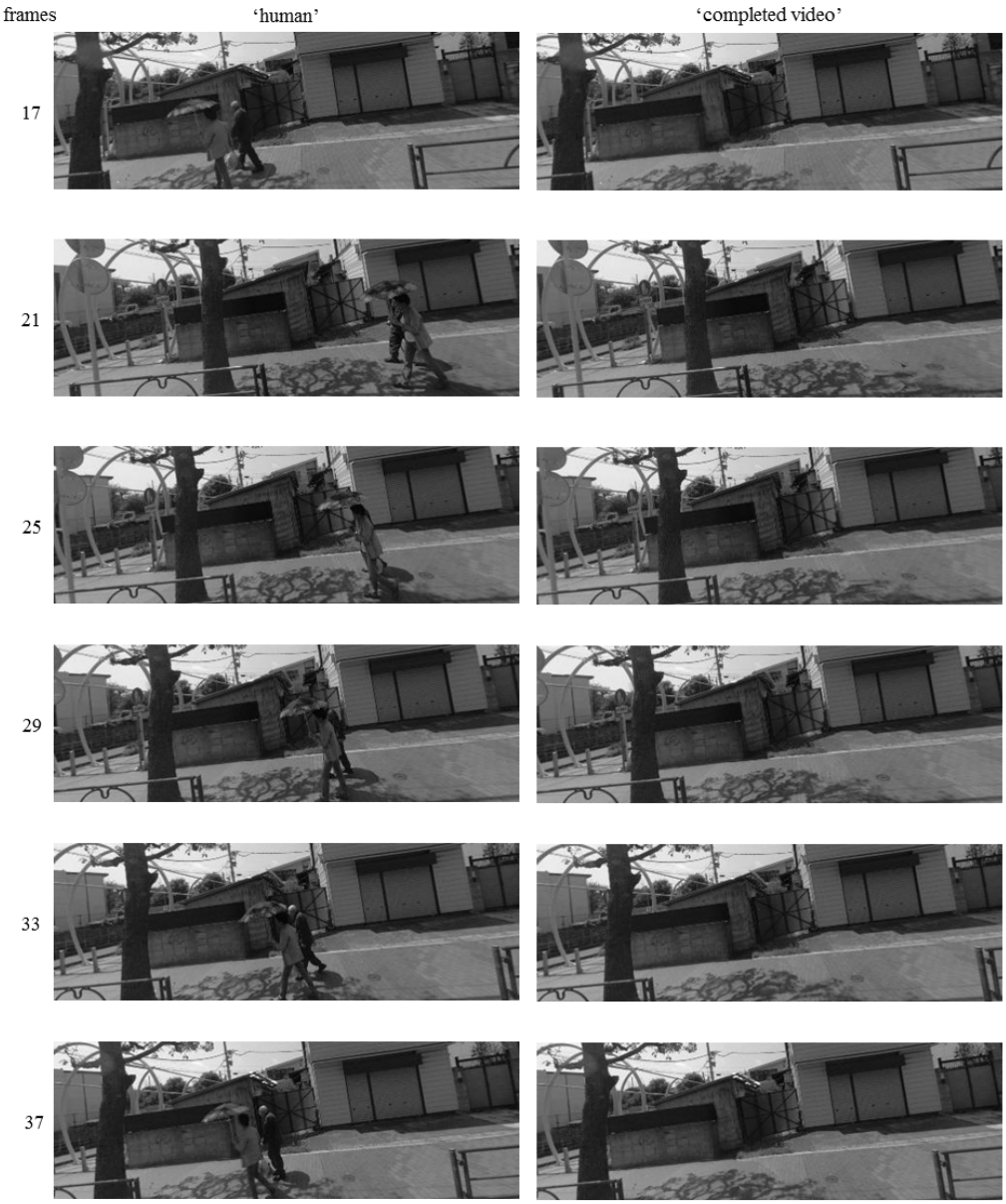

Figure 5.26: Representative frames of the result of completion of a real street video where the pedestrians are removed.

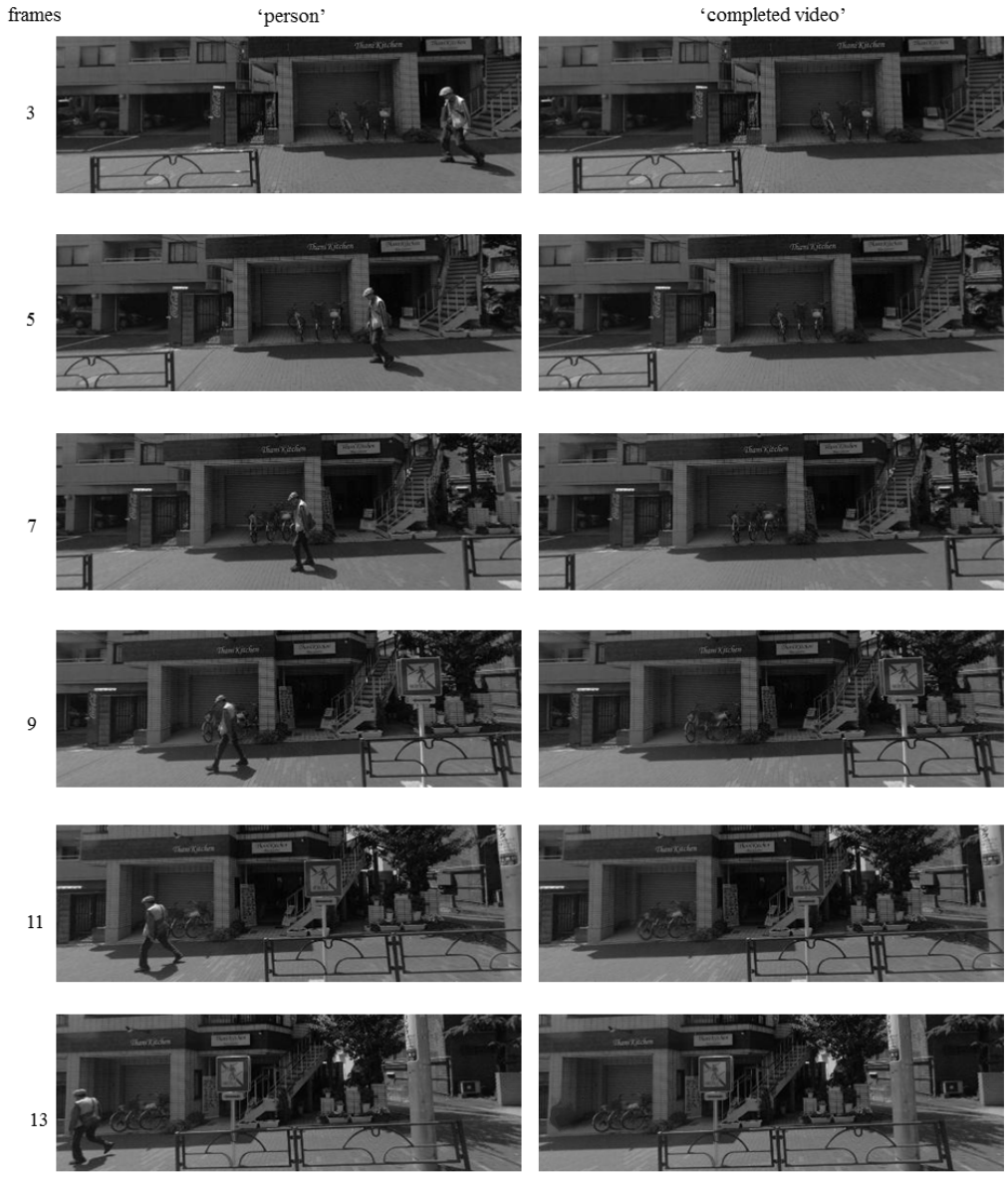

Figure 5.27: Representative frames of the result of completion of a real street video where the pedestrians are removed.

CHAPTER

SIX

Refining Boundaries of Inpainted Motion

Contents

In the previous chapters, we have presented a spatio-temporal motion inpainting and color propagation method. So far, we have assumed that the motion of the hole and its boundary is smooth in both spatial and temporal domains. In actual applications, this assumption is not always correct. The motion of an object is, in fact, restricted within the object itself and therefore smooth motion is violated along its boundaries (i.e. the motion between a building and the sky, a pole and a planar background etc.)

In cases where the inpainted hole contains more than one object that have different depth with respect to the camera, there is a need to limit the effect of the smoothness constraint along the boundary of these objects. In traditional

motion inpainting techniques where the optical flow is corrected in occluded regions, this is done by inferring the correct value of the flow from its color and the color of the neighboring pixels. If their colors are very close to each other, the optical flow is therefore mostly related to one another.

Similarly, the edges of objects gives some information on the motion boundary. Almost all the time, a motion boundary corresponds to an object boundary (the converse is not true). It is possible, however, to estimate the motion boundary by assuming that the optical flow is discontinuous along the object boundaries. In effect, if this boundary is in fact a motion one, the estimated optical flow will be highly accurate.

Obviously, the motion inpainting techniques for estimating the optical flow in occlusion regions, and the use of edge information to estimate the motion boundaries, will not work when applied to video completion because neither the color nor the edge information is available inside the hole. In Chapter 5, we proposed to gradually include the newly inpainted color inside the hole after one iteration of the color propagation back to the brightness constraint of the optical flow. In this manner, we were able to refine the result of the video completion. However, this method is still problematic because the initial values solved (the first optical flow estimated when the hole is still empty) is basically spatio-temporally flat. This means that the method in Chapter 5 works well in cases where the inpainted background is smooth or the difference of motion along boundaries is not very large.

In this chapter, we first propose to improve the initial estimated value of the optical flow inside the hole by guessing the edge information and therefore limit the spatial smoothness of the motion. We estimate the edges in the hole by transferring the edges from the known parts of the video following an affine transformation based on the optical flow along the hole boundary. We show an improvement in the results of the spatio-temporal motion inpainting in Chapter 4.

This chapter is organized as follows. First, we will present our edge transfer method that is based on tracking. Then, we will modify the spatio-temporal optical flow estimation and inpainting framework to accommodate the changes induced by usage of edge information. After that, we will show how the improved motion boundary poses a problem in color propagation in lower resolution of the image pyramid due to undetected occlusion regions and present a method to solve this problem. Finally, we show some experimental results followed by the chapter summary.

6.1 Related Work

6.2 Optical Flow-Based Edge Transfer Method

Suppose we have an r^{th} frame I_r with hole H_r and edge information represented by a binary label $D_r = D_{r_H} + D_{r_\Omega}$, where D_H and D_Ω are the edges inside and outside the hole, respectively. Our goal is to find an estimate of D_{r_H}. Since we do not have any color inside the hole, it is necessary to solve for D_{r_H} by basing it on the edge information from the known regions in other frames $D_\Omega \setminus D_{r_\Omega}$.

We first estimate the boundaries of objects in the whole sequence using Canny edge detector. We label the detected edges as $\{0 = edge, 1 = non - edge\}$. We also run an initial estimate of the optical flow u of Ω.

Then, we select the region of points p_r along the boundary of H_r and track these points p_s in frame I_s by following u, assuming that the desired value of H_r from is completely visible in I_s. This can be successfully done through proper inference from u. We select the highest velocity (in pixels/frame) u_{p_r} in the region p_r and the actual velocity of the hole u_{H_r}. Assuming that the hole does not change in diameter, we let $\oslash(H_r)$ as the diameter of H_r. If we also assume that the velocity of p_r and H_r are constant, the temporal distance, $(r - s)$, between frames I_r and I_s can be solved using Newtonian relativity. That is:

$$r - s = \frac{\oslash(H_r)}{u_{H_r} - u_{p_r}} \tag{6.1}$$

Since the points p_r and p_s correspond to different objects, they will eventually be scattered according to the movement of the object they belong to. If the points belong to one object, we can assume that these points will move in a relatively similar manner. With that said, we can cluster all the tracked points according to their optical flow values to segment p_s into different objects. That is, $p_s = \sum_{k=1}^{j} c_k$ for j number of objects.

The problem lies in choosing which cluster to follow that will transfer the appropriate edge information to the hole. Since we have clusters belonging to different objects with varying depths, it is possible that the edges that contains the object will vanish due to occlusion from other objects. With this in mind, however, we can say that edges belonging to foreground objects (or lesser depth) will have a higher probability of being preserved.

With this assumption, we arrange the clusters according to their optical flow value (object velocity). At this point, we will assume that the motion of objects in the video are largely due to the camera motion, and therefore we can say that objects with higher velocity will have lesser depth (or closer to the camera).

After choosing a cluster, say c_s, we then roughly estimate the size of the region H_r^s (same size as the hole in I_r in our implementation) that will be transfered from I_s. We solve the transformation matrix between c_s and c_r using affine

Sequence	Error (Average Distance)	No. of missed pixels
rubberwhale	0.240	47
urban3	0.561	32
urban2	0.838	68

Table 6.1: Difference in the estimated edge and actual edge measured using the distance and the difference in the number of detected pixels.

transformation assumptions:

(6.2)

Using this transformation, we transfer the edges in H_r^s to H_r. We illustrate the whole process in Figure 6.1.

We test this method using several image sequences and we show the results in Figure 6.2 and 6.3. We also compare the estimated edge on the actual edge data and show that our result is very close to the actual value. The findings are summarized in Table 6.1 for the Middlebury set in Figure 6.2.

6.3 Refining the Boundaries of Motion Inpainting

In this section BLah blah blah

6.3.1 Modifying the Spatial Term of the Optical Flow

Now that we have estimated the edge inside the hole, it is still necessary to modify the spatial smoothness constraint of the motion inpainting method presented in Chapter 4. We have found out in our experiments that variational optical flow methods, as we have implemented in this work, have problems when it comes to varying the spatial smoothness constraint (or the regularizer in general).

Variational methods have a large dependency on the median filtering step after every iteration in the estimation process (SEE APPENDIX). Sun ET AL proposed to implement a weighted non-local regularizer in addition to the general TV spatial term to incorporate the median filter in the optimization function as:

$$E_{spatial} = \min_{u,\hat{u}} \psi_{TV}(\nabla u) + \frac{1}{2}\|u - \hat{u}\|^2 + \sum_{k}^{N} w_k \, |\hat{u}_i - \hat{u}_k|_2^2 \tag{6.3}$$

However, in their method, it appears that they still use the output of the median filter to be the final value. We found out, however, that the median filter in the final step is actually necessary because another run of the iteration

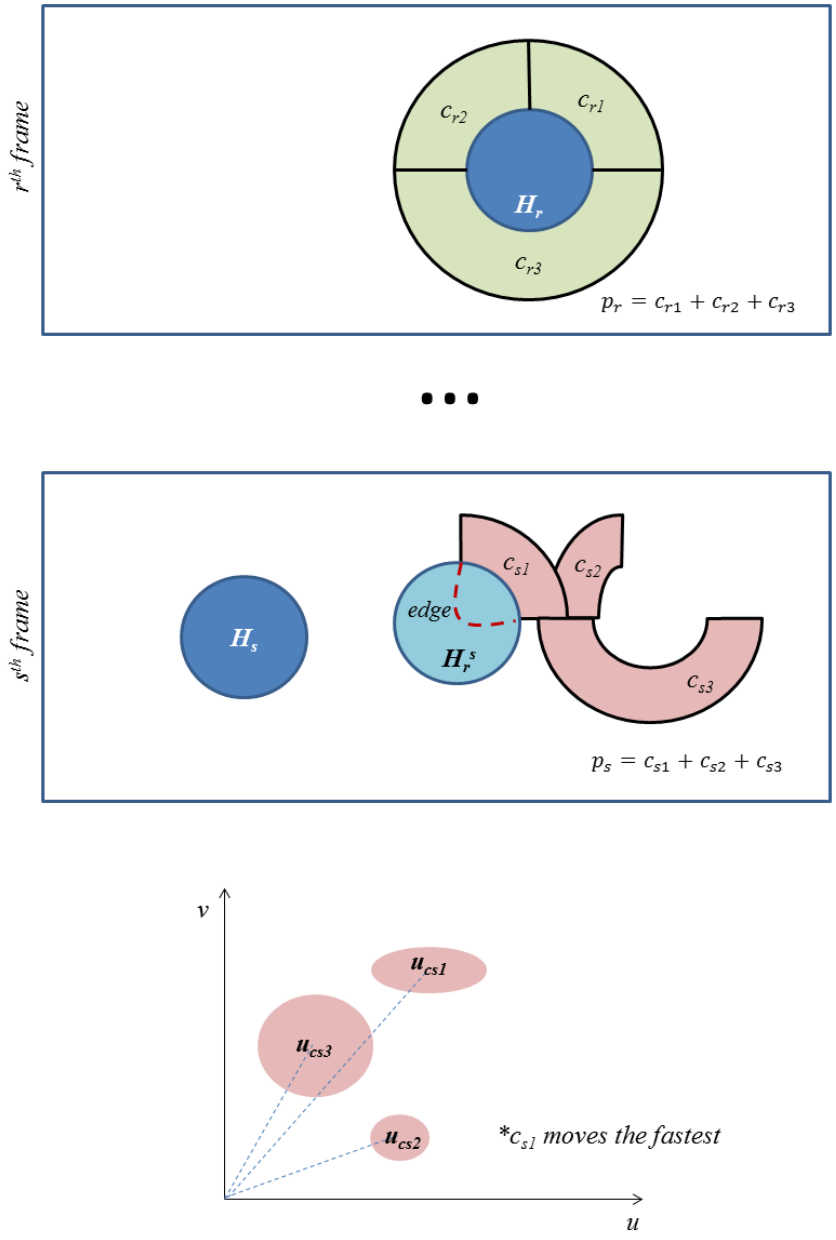

Figure 6.1: Edge transfer method.

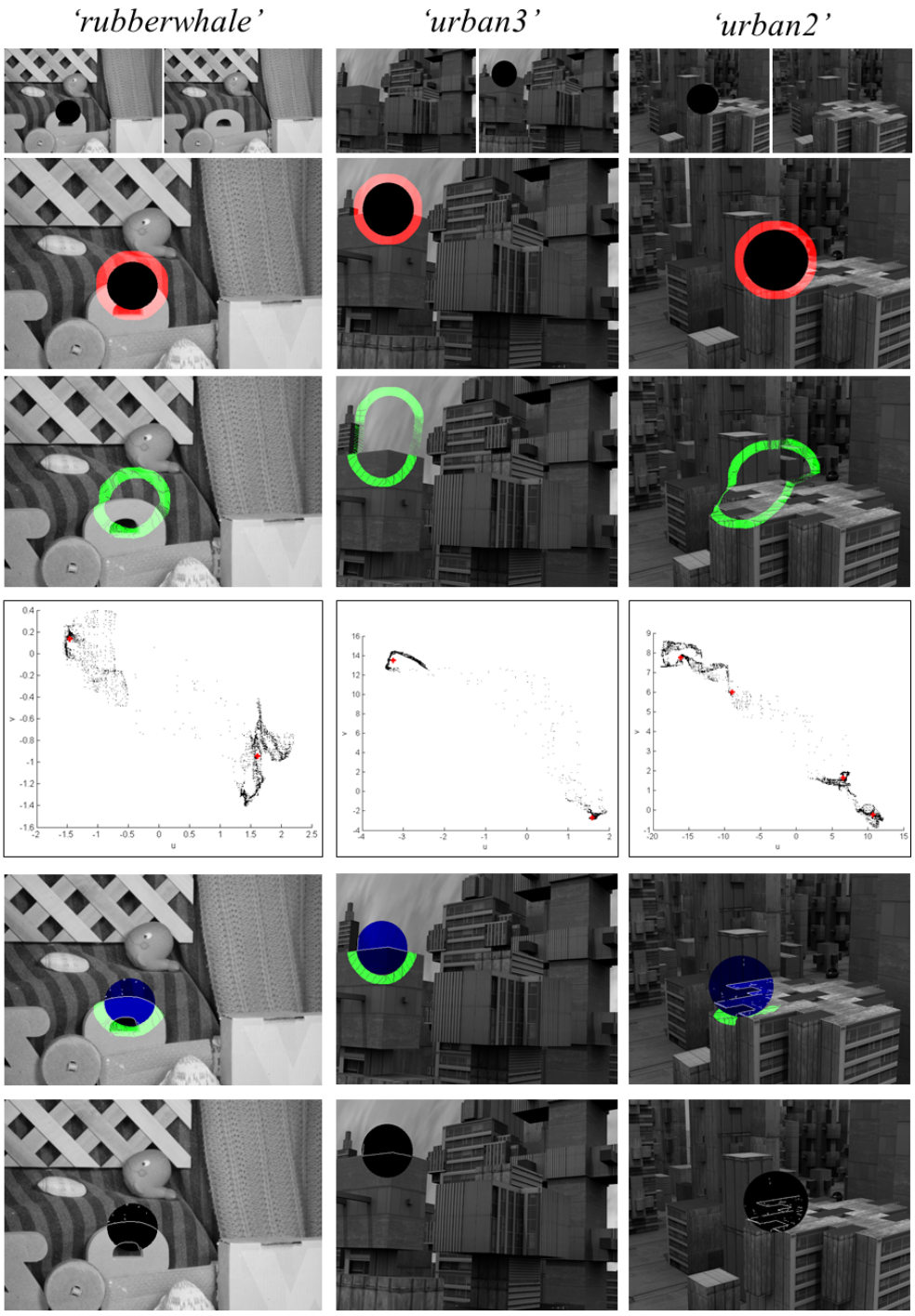

Figure 6.2: Result of edge transfer method on Middlebury dataset. **From left to right**: rubberwhale', 'urban3' and 'urban2'. **From top to bottom**: 1) reference and source frame, 2) selected points around the hole in the reference frame (red), 3) tracked points in the source frame (green), 4) clustering based on optical flow of the points, 5) selected cluster in the source frame with highest velocity (green) end estimated hole position (blue) with detected edge to be transferred, and 6) transfered edges in the hole.

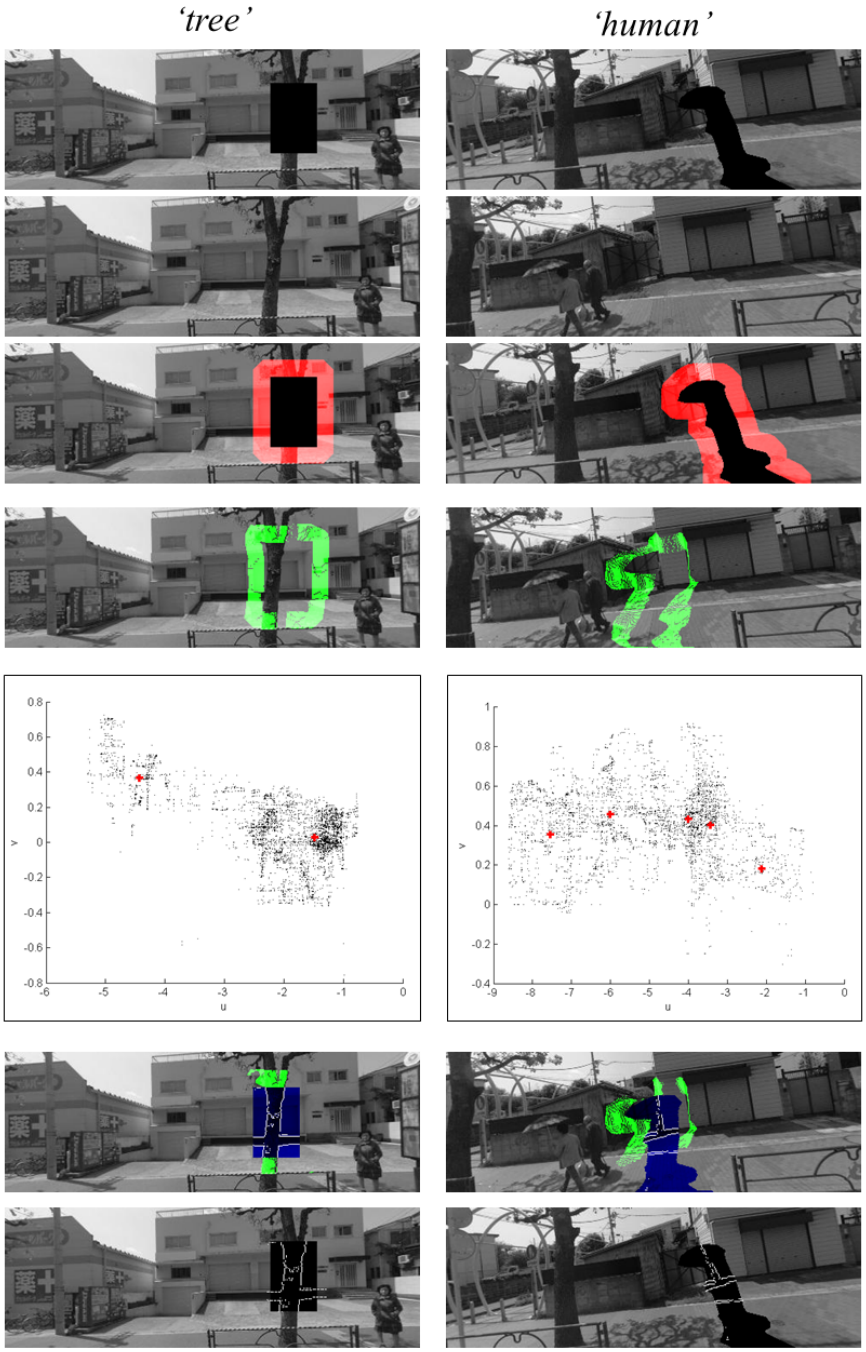

Figure 6.3: Result of edge transfer method on real dataset. **From left to right**: 'tree', 'human'. **From top to bottom**: 1) reference, 2) source frame, 3) selected points around the hole in the reference frame (red), 4) tracked points in the source frame (green), 5) clustering based on optical flow of the points, 6) selected cluster in the source frame with highest velocity (green) end estimated hole position (blue) with detected edge to be transferred, and 7) transfered edges in the hole.

Sequence	End-Point Error		Runtime (s)	
	Sun et al.	Our method	Sun et al.	Our method
rubberwhale	0.180	0.187	135.9	120.1
hydrangea	0.339	0.350	138.4	123.2
grove2	0.175	0.179	186.5	180.5
grove3	0.747	0.744	186.7	126.3
urban2	0.854	0.909	183.3	177.4
urban3	0.874	0.831	209.2	123.6

Table 6.2: TV+median [Sun *et al.* 2010] vs. median only regularizer. The small decrease in the end-point error, which is almost negligible, is a better trade-off for a faster and more efficient solution of the optical flow estimation.

gives far more outliers without it. We also found out that using the median filter at the end of every iteration neglects the effect of the TV term. In this effect, even if the $E_{spatial}$ is modified using a spatially varying value, such as $\lambda_s(x)E_{spatial}$, where $\lambda_s(x)$ is based on the edge information, the result will still largely depend on the median filter. Hence, if the median filter is not spatially varied as $E_{spatial}$, the resulting optical flow will not have a discriminating effect on the object boundaries.

With this in mind, we simplify the method presented by [Sun *et al.* 2010] and completely remove the TV term. As a result, we have

$$E_{spatial} = \min \frac{\lambda_m}{2} \|u - \hat{u}\|^2 + \sum_{k}^{N} w_k \, |\hat{u_i} - \hat{u_k}|_2^2 \tag{6.4}$$

We compare the optical flow estimation results of using (6.3) and (6.4) and show the sample outputs in Figures 6.4 and 6.5 and the end-point error and running time in Table 6.2. There results show that there is no significant decrease in the end-point error without using the TV spatial term. However, we were able to consistently improve the iteration time by using only the median filter as spatial constraint.

6.3.2 Motion Estimation and Inpainting

Following the method in Chapter 4, we perform a joint motion estimation and inpainting algorithm. With the changes in the previous section, the new opti-

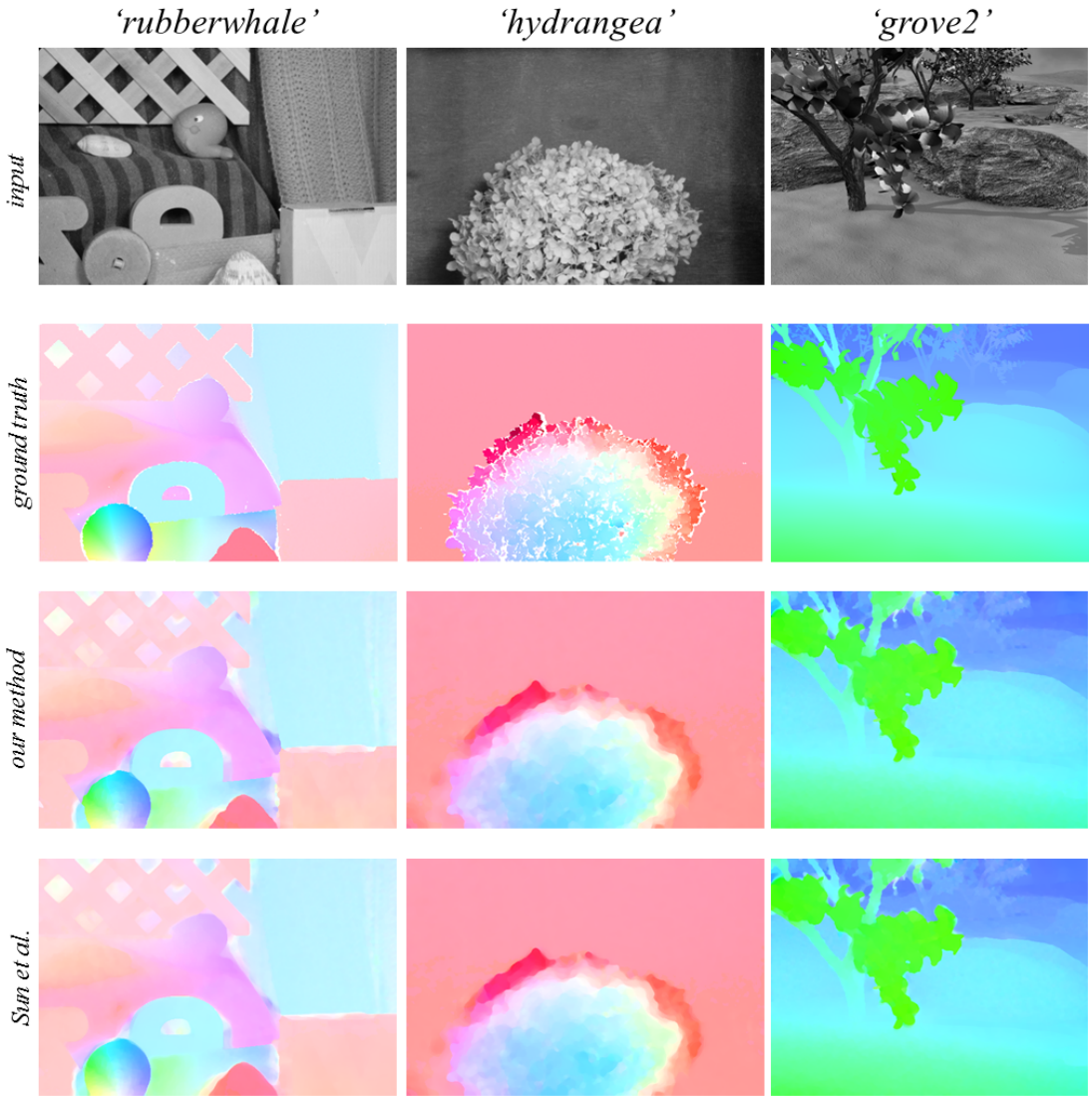

Figure 6.4: TV+median [Sun *et al.* 2010] vs median only regularizer.

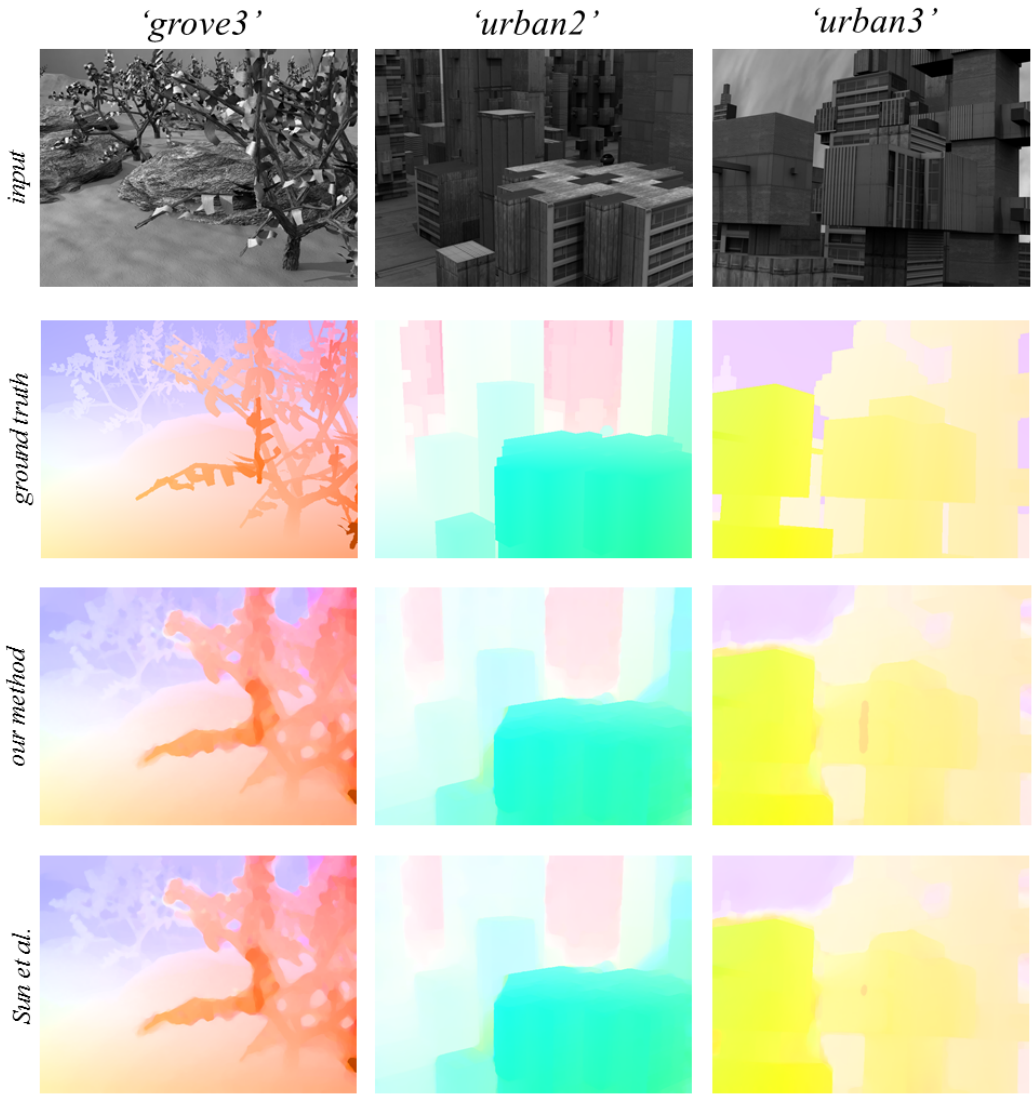

Figure 6.5: TV+median [Sun *et al.* 2010] vs median only regularizer.

mization function we need to solve is:

$$\min_{u_f,u_b} \lambda_d \left[\psi\left(I_f(x+u_f)-I_0\right)+\psi\left(I_b(x+u_f)-I_0\right)\right]$$

$$+\frac{\lambda_m}{2}\|u_f-\hat{u_f}\|^2+\sum_k^N w_k\left|\hat{u_{f_i}}-\hat{u_{f_k}}\right|_2^2 \tag{6.5}$$

$$+\frac{\lambda_m}{2}\|u_b-\hat{u_b}\|^2+\sum_k^N w_k\left|\hat{u_{b_i}}-\hat{u_{b_k}}\right|_2^2 \tag{6.6}$$

$$+\frac{\lambda_t}{2}\left|\phi(u_f,u_b)-b^k\right|_2^2+const. \tag{6.7}$$

The estimated edge inside the hole D_H is dilated and graduated by using a 3x3 box filter. This expands the edge by one more pixel in order to eliminate the boundary during median filtering. We use the edge value in as the weighting w_k.

To minimize 6.5, we perform the following ADMM technique. First we hold u_b and b^k constant and minimize:

$$\min_{u_f,\hat{u_f}} m(x)\left|I_f(x+u_f)-I_0(x)\right|+\lambda_s\|u_f-\hat{u_f}\|^2+\sum_k^N w_k\left|\hat{u_{fi}}-\hat{ufk}\right|+\lambda_t\left|\psi(u_f,u_b)-b^k\right| \tag{6.8}$$

Again, to solve for the above function, we alternately minimize between u_f and $\hat{u_f}$:

$$\min_{u_f} m(x)\left|If(x+u_f)-I_0(x)\right|+\lambda_s\|u_f-\hat{u_f}\|^2+\lambda_t\left|\left|\psi(u_f,u_b)-b^k\right|\right|+const. \tag{6.9}$$

$$\min \hat{u_f} m(x)\lambda_s\|u_f-\hat{u_f}\|^2+\sum_k^N w_k\left|\hat{u_{fi}}-\hat{ufk}\right|+const. \tag{6.10}$$

The solution for u_f can be found by solving the Euler-Lagrange equations for both directions and performing a simple point-wise algebraic manipulation. For $\hat{u_f}$, the solution is proposed in [Li & Osher 2007]. The same is done for u_b and the b^k is updated as $b^{k+1}=\phi(u_f,u_b)-b^k$.

6.4 Improvement of Simultaneous Motion Inpainting and Color Propagation Method

The method discussed in the previous section can be further extended to the method described in Chapter 5. We use the optimization function in 6.5 instead

and perform the Algorithm 3. The improvement in the method is two-folds. First, the initial inpainted motion used for the first iteration of the color propagation is more accurate and therefore, the convergence to a better solution can be achieved. Second, the combined edge refinement strategy and the spatially varying mask function will further improve the resulting estimated motion.

There are several assumptions that needed to be discussed here. First, one can say that a wrong estimation of the edge information will instead degrade the quality of the estimated motion rather that improve it. We can argue that, however, the effect of the spatial smoothness will have to be lessen just enough to have the data and trajectory constraint to be stronger.

By dilating the area of the edges (instead of just a strict 1 pixel wide line), we can expand the reduction effect on the spatial smoothness of the flow. This will have two effects on the inpainted motion, if ever we made a wrong guess. First, if an edge is wrongly placed on a supposedly smooth area, the spatial smoothness constraint will in fact be lessen. However, since there is no real boundary in the motion, the data and trajectory constraint will not have greater influence on the estimation than the spatial smoothness. Therefore, the optical flow will be estimated as if there is no boundary there.

Secondly, if no edge is placed on a real motion boundary, the quality of the inpainting will just be similar to the one without the edge transfer method. In other words, the improvements in the inpainting results will only be increased if the estimated edge lies correctly on a real motion boundary, and will have no effect otherwise.

6.5 Occlusion Handling during Color Propagation

With the improvement of the edges of the inpainted motion, the problem of occlusion become more apparent. This problem is due to color and motion ambiguity. We illustrate this problem in Figure 6.6. Say, we are inpainting the hole in frame 2 which is located at the background and project very near to the boundary of the foreground (gray cylinder). Assuming that we get a very accurate motion of the hole through motion inpainting, we then propagate the colors from both directions (frames 1 and 3). For frame 1, the backward flow points to a point in the background which is correct. In frame 2, the forward flow points to the foreground region which is wrong.

With the previous color propagation technique described in Chapter 5, the two values will be averaged since they are only 1 frame distance away from frame 2 ($\mu = 1$). Therefore, if we look at its inpainted color, we will see a smoothed version of the foreground and the background. This effect is shown in Figure 6.7.

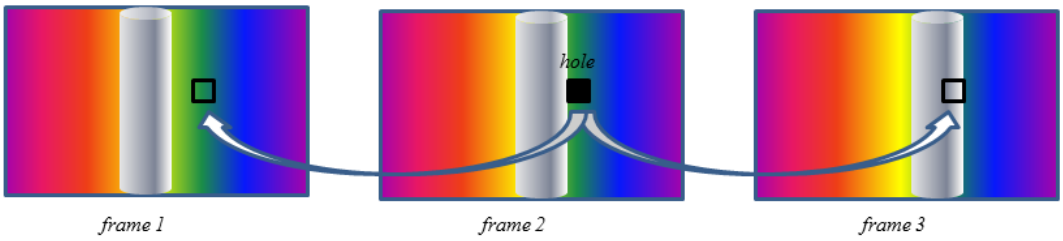

Figure 6.6: Color and motion ambiguity.

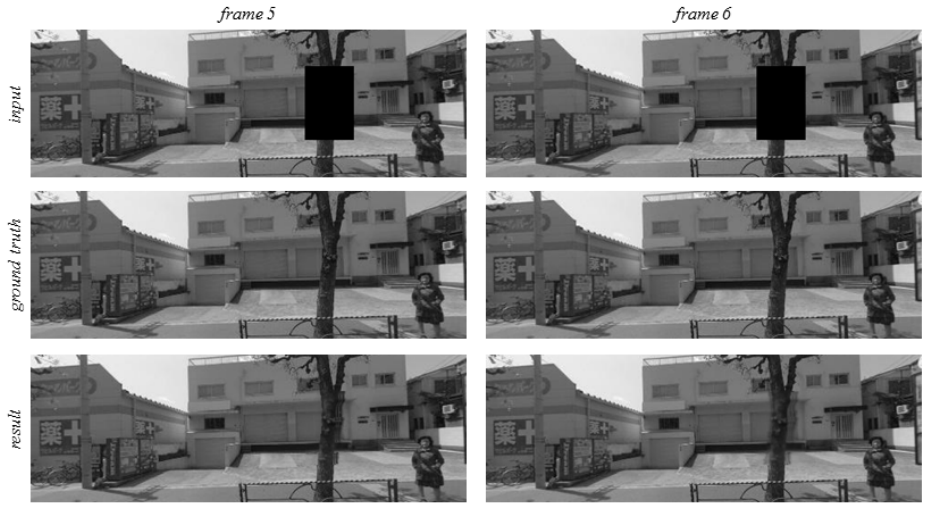

Figure 6.7: Degradation in the inpainted color along the boundary of the foreground and the background region due to color and motion ambiguity.

We solve this problem by simple reasoning on the order of propagation. By using the intermediate inpainted motion, we estimate the occlusion region in all frames. If the motion points to this region, we ignore this value and find a different value from other frames. Most of the time, this will be compensated on both direction and therefore only occlusion region that creates ambiguity between equidistant frames are needed.

6.6 Experimental Results

In this section we show the final result of the refined edge transfer and compare and contrast the results obtained from the previous chapter.

6.6.1 Comparing Edge Refined Inpainting

We first compare the inpainting results obtained by using the method in Chapter 5. We chose a part of a street video where there is a foreground tree with a well defined texture and a background with textureless facade of a building. The case that we want to emphasize is that when two depths in the hole are inpainted, there will be an oversmoothing in the resulting inpainted motion. This happens especially if the background is without texture and therefore its optical flow is hard to estimate. As a result, the values tend to be interpolated on the more textured foreground which results in inpainting error.

The comparison between the inpainted motion (and result of optical flow in general) is shown in Figure 6.8. Before the improvement in the edge estimation of the optical flow, the motion of the background and the foreground is mixed along the boundary. Moreover, the boundaries of motion in other parts of the video is not very well defined. We then perform the whole process in Chapter 4 and was able get a result with better quality. We show this in Figure 6.9. Note that in the result of the method before edge refinement, the background building seems to move with the tree. If we look closely, the details are distorted near their boundaries. In the improved method, the background was successfully inpainted because its motion was accurately estimated. The remaining problem (averaging of tree and building color) in occlusion regions is addressed by the improved color propagation and the result is shown in the next section.

6.6.2 Comparing Improve Color Propagation with Occlusion Handling

We now compare the results of edge refined motion between the improved color propagation with occlusion handling and the one without it. We use the previous image sequence and perform the iterative inpainting and propagation on both

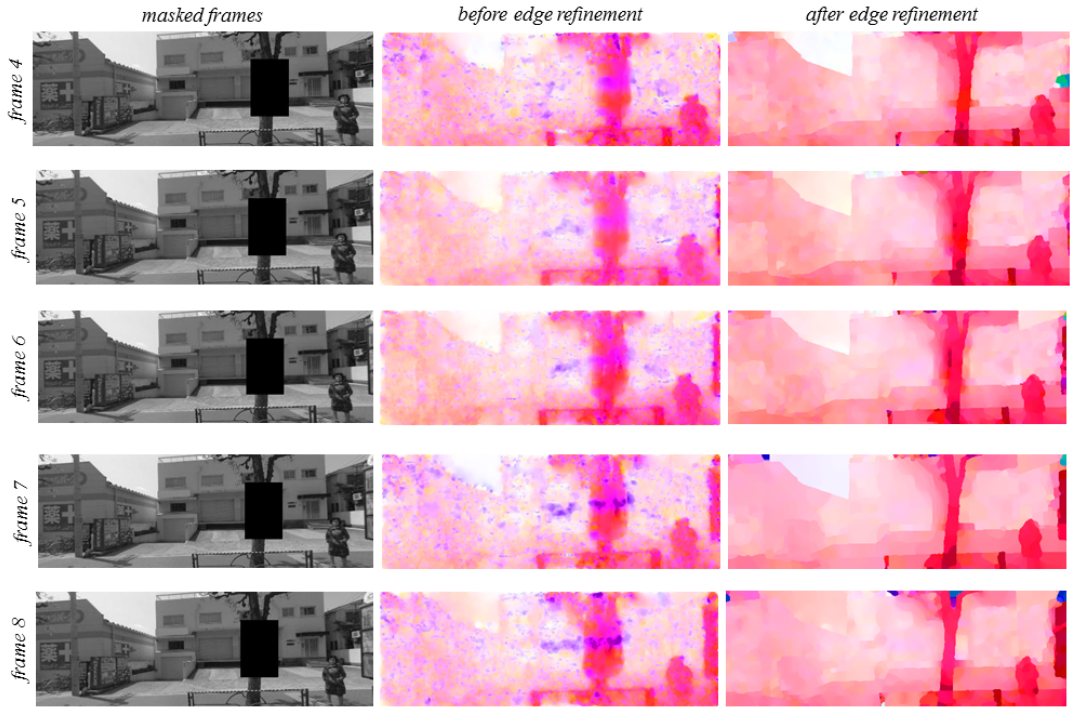

Figure 6.8: Comparison of the motion inpainting with and without the edge refinement method.

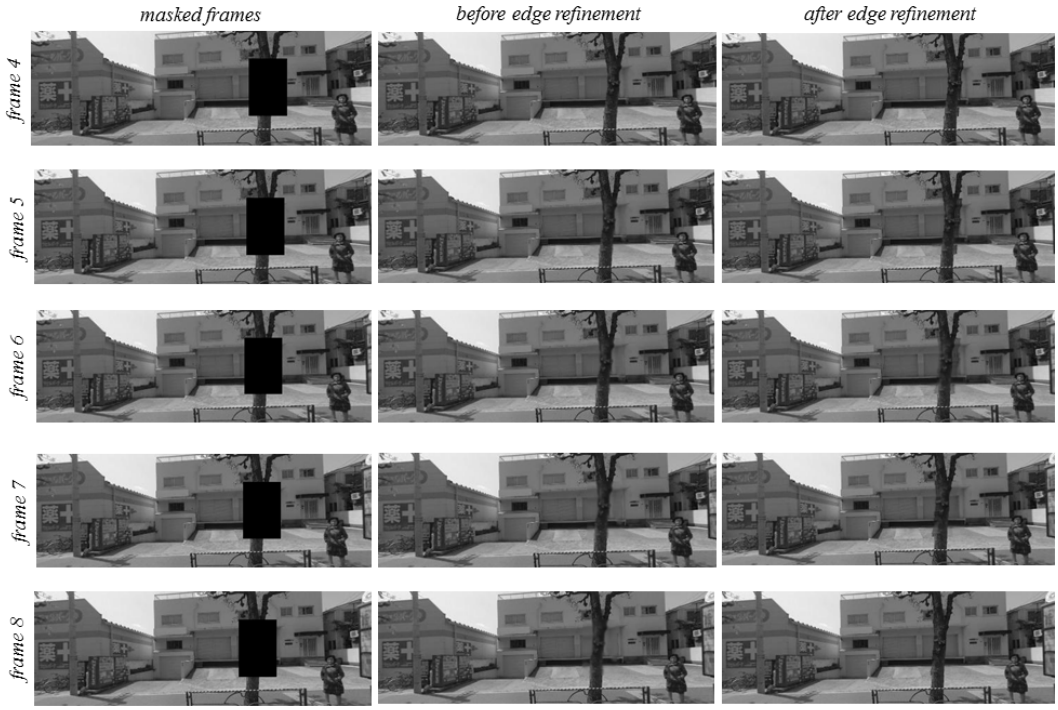

Figure 6.9: Comparison of the completed video with and without the edge refinement method. The color propagation used in this results does not have occlusion handling.

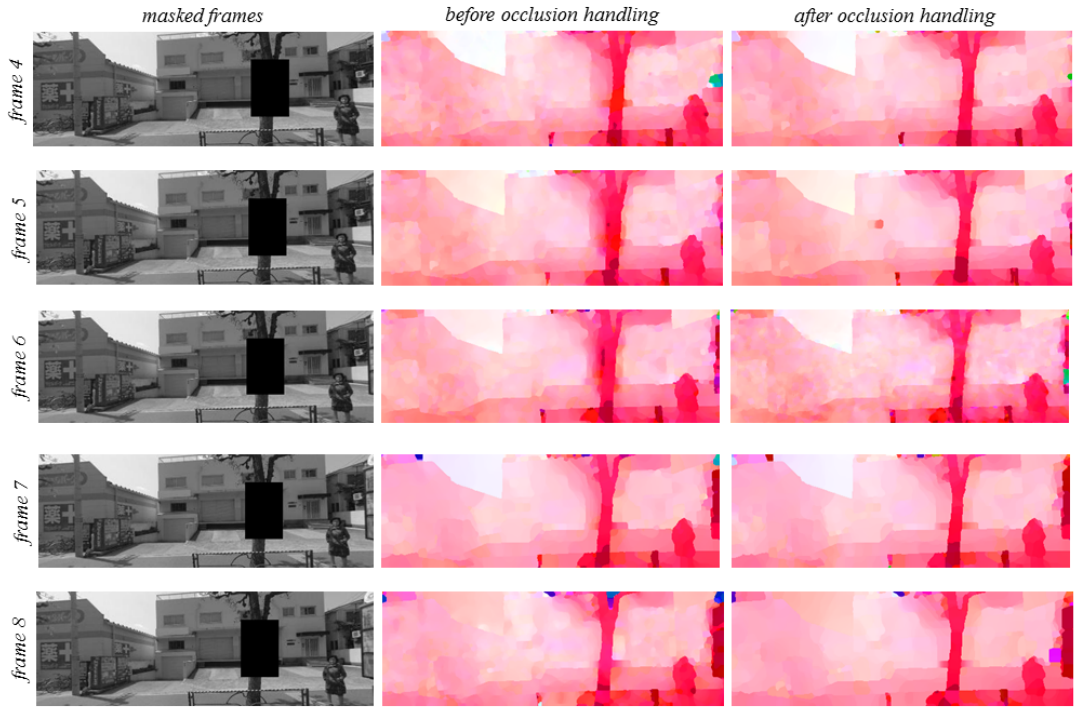

Figure 6.10: Results of motion inpainting after occlusion handling improvement of the color propagation. The inpainted motion is also improved.

techniques. The improvement in the motion inpainting is shown in Figure 6.10 and the resulting video in Figure 6.11.

The improvement in the output is two-folds. First it is obvious that with the occlusion handling, the averaging effect will be removed along the boundary of the background and foreground as shown in the results. Second, since the color propagation output is improved, when we perform the iterative approach, the improved estimated color also improves the inpainted motion. In other words, we were able to remove the errors in motion inpainting that was caused by the wrong color propagation results.

6.7 Chapter Summary

In this chapter, we proposed a boundary refining method on the inpainted motion inside the hole by estimating the edges in it and by using a spatially weighted median regularizer. Since there is no color information in the hole, we proposed an edge transfer method that copies the edges from known frames by following an affine transformation assumption. We computed the affine transformation by using the known points along the boundary of the hole and compare them with the points in the frame with known edges.

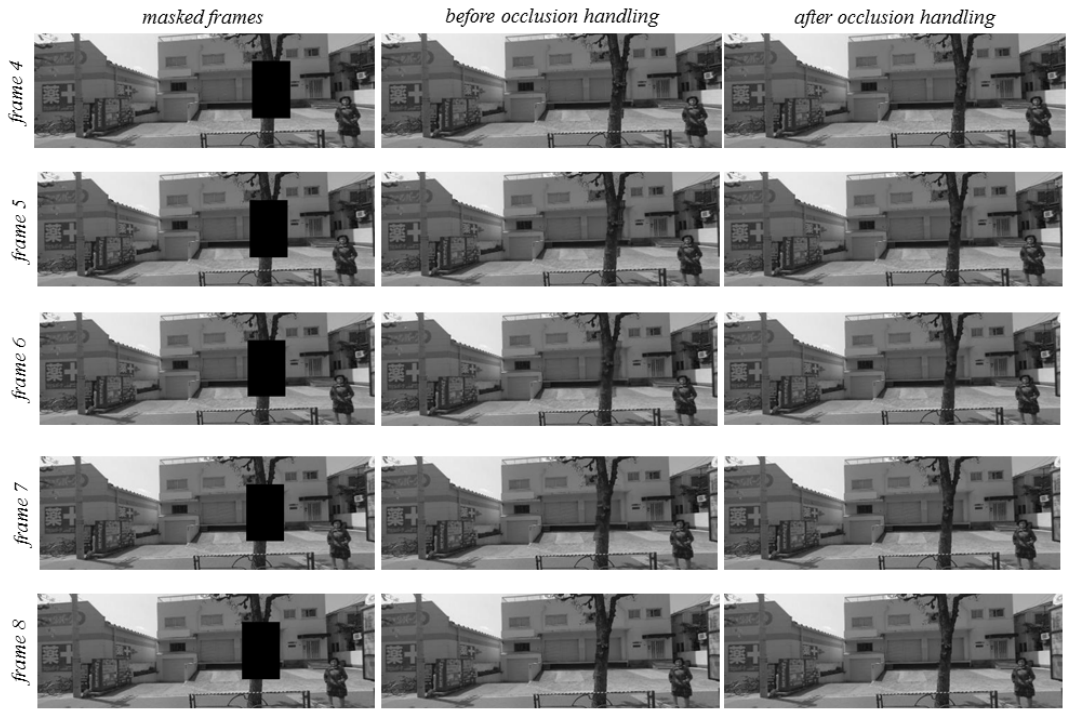

Figure 6.11: Results of video completion after occlusion handling. The ambiguity was completely removed along the boundary of the foreground and the background.

We also improved the color propagation technique to handle occlusion problems that occurred after the improvement in the motion boundaries. We then tested our results in both synthetic and real videos. We showed that our method can handle difficult problems such as the presence of multiple depths (foreground and background) which is handled in other works using layering and user assistance. Compared to these methods, our work is fully automatic and does not require detection of a foreground and a background region.

CHAPTER

SEVEN

Conclusion

7.1 Summary

In this thesis we proposed to solve the video completion problem by using a spatio-temporally consistent motion inpainting. To summarize, the contribution of our work are three-folds. First, we proposed a framework in inpainting motion using multiple frames by imposing a smooth spatial and trajectory constraint on the motion among the frames. We did this using a joint motion estimation and inpainting algorithm that utilizes a binary label mask to eliminate the effect of the color information inside the hole. The smoothness constraints proved to be effective in propagating the known motion from the boundaries towards the hole. We tested this method, compared them with prior work and showed that our method works better in inpainting motion.

Secondly, we proposed a simultaneous motion inpainting and color propagation method by using an iterative optimization method. We obtained better results when we used the newly inpainted pixels inside the hole to refine the optical flow estimation inside it. We control the effect of the newly inpainted pixels using our proposed mask function that relates the frame distance of the source pixel to the reference pixel in the hole. We also introduced a trajectory prior estimation method to handle the trajectory constraint during non-smooth motion. Our method comprised of only three frames and therefore was implemented really fast. We also improved the standard color propagation method to include a technique in combining the result of two directions, namely the forward and the backwards. We combined the propagated color from both directions using our proposed blending technique. We then showed in our result that this method can accomplish video completion results accurately.

Finally, we proposed a boundary refining method on the inpainted motion inside the hole by estimating the edges in it and by using a spatially weighted median regularizer for motion estimation. We estimated the edge inside the hole

by transferring the known edges from other frames to the reference frame using affine transformation which is based on the motion of the surrounding pixels. We then again improved the color propagation technique to handle occlusion problems that occured after improving the boundary of the inpainted motion. We showed in our results that this proposed method works very well even if there are multiple depths in the hole.

7.2 Future Direction

Our optimization framework is designed to be extended to virtually any optical flow estimation method. A choice of a good functional will result in faster approximation of the motion inside the hole and eventually faster video completion results. A real-time implementation is also desired since a lot of application, such as mixed reality, require that an object is removed and inpainted in real-time. Since the motion can be estimated by using only three frames, a real-time implementation is very possible. The only limitation is when the hole extends several frames that the first available source pixel is very far from the current frame.

A desired extension is to combine the masking of the hole and the inpainting method proposed in this thesis into one automatic framework. The burden is put on the detection and tracking of the unwanted object on all the frames in real-time.

Another possible improvement is to modify the constraints to handle more dynamic motion such as those of non-rigid objects and to consider more complex scenes such as places with clutters.

We conclude this thesis by saying that video completion is a very hard task and requires a lot of engineering in order to be useful in most applications. However, with the emergence of fast computers and algorithms, this problem is not really far from being perfectly solved.

BIBLIOGRAPHY

[Baker *et al.* 2007] S. Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black and Richard Szeliski. *A Database and Evaluation Methodology for Optical Flow*. In ICCV, 2007.

[Bao *et al.* 2014] L. Bao, Yang Q. and Jin H.s. *Fast edge-preserving patch match for large displacement optical flow*. In CVPR, 2014.

[Black & P. 1991] M. Black and Anandan P. *Robust dynamic motion estimation over time*. In In proc. CVPR, 1991.

[Black & P. 1996] M. Black and Anandan P. *The robust estimation o fmultiple motions - parametric and piecewise smooth flow fields*. In CVIU, 1996.

[Bruhn *et al.* 2005] A. Bruhn, Joachim Weickert and Christoph Schnörr. *Lucas/Kanade meets Horn/Schunck - combining local and global optic flow methods*. IJCV, vol. 61, no. 3, pages 211–231, 2005.

[Burt & Adelson 1983] P.J. Burt and E.H. Adelson. *The laplacian pyramid as a compact image code*. IEEE Transactions on Communication, vol. 31, no. 4, pages 532–540, April 1983.

[Burt 1981] P.J. Burt. *Fast filter transforms for image processing*. In Computer Graphics and Image Processing, 1981.

[Burt 1983] P.J. Burt. *Fast algorithms for estimating local image properties*. In Computer Graphics and Image Processing, 1983.

[Butler *et al.* 2012] D. J. Butler, Jonas Wulff, Garrett Stanley and Michael J. Black. *A naturalistic open source movie for optical flow estimation*. In ECCV, 2012.

[Chan *et al.* 2005] T. Chan, Andy M. Yip and Frederick E. *Simultaneous Total Variation Image Inpainting and Blind Deconvolution*. Int'l Journal of Imaging Systems and Technology, vol. 15, no. 1, pages 92–102, 2005.

[Chen & Lorenx 2011] K. Chen and D. Lorenx. *Image Sequence Interpolation using Optimal Control*. Journal of Mathematical Imaging and Vision, vol. 41, pages 222–238, 2011.

[Chen *et al.* 2014] C. Chen, He B., Ye Y. and Yuan X. *The direct extension of ADMM for multi-block convex optimization problem is not necessarily convergent*. Mathematical Programming, 2014.

[Criminisi *et al.* 2003] A. Criminisi, P. Perez and K Toyama. *Object removal by exemplar-based inpainting*. In CVPR, 2003.

[Dahl *et al.* 2010] J. Dahl, Per Christian Hansen, Søren Holdt Jensen and Tobias Lindstrøm Jensen. *Algorithms and Software for TV Image Reconstruction via First Order Methods*. Numerical Algorithms, vol. 53, pages 67–92, 2010.

[Goldstein & Osher 2009] T. Goldstein and S. Osher. *Split Bregman Method for L1-Regularized Problems*. SIAM Journal on Imaging Sciences, 2009.

[Granados *et al.* 2012a] M. Granados, W. Tai-Pang, Yu-Wing Tai and Chi-Keung Tang. *Background Inpainting for Videos with Dynamic Objects and a Free0moving Camera*. In Proceedings fo the European Conference on Computer Vision, 2012.

[Granados *et al.* 2012b] M. Granados, J. Tompkin, K. Kim, O. Grau, J. Kautz and C. Theobalt. *How Not to Be Seen - Object Removal from Videos of Crowded Scenes*. Computer Graphics Forum, vol. 31, no. 2.1, pages 219–228, 5 2012.

[Hartley & Zisserman 2004] R. I. Hartley and A. Zisserman. Multiple view geometry in computer vision. Cambridge University Press, ISBN: 0521540518, second édition, 2004.

[Horn & Schunck 1981] Berthold K. P. Horn and Brian G. Schunck. *Determining Optical Flow*. ARTIFICAL INTELLIGENCE, vol. 17, pages 185–203, 1981.

[Jia *et al.* 2004] J. Jia, Others and Others. *Video Repairing: Inference of Foreground and Background Under Severe Occlusion*. In CVPR '04 Proceedings, 2004.

[Jia *et al.* 2005] Y. Jia, Shi-Min Hu and Ralph R. Martin. *Video Completion Using Tracking and Fragment Merging*. The Visual Computer, pages 601–610, 2005.

[Krotkov 1989] Eric Paul Krotkov. Active computer vision by cooperative focus and stereo. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1989.

[Li & Osher 2007] Y. Li and S. Osher. *A new median formula with applications to PDE based denoising*. Communication Mathematics Science, vol. 7, no. 3, pages 741–753, 2007.

[Li *et al.* 2012] X. Li, Jia J. and Matsushita Y. *Motion detail preserving optical flow estimation*. TPAMI, vol. 34, no. 9, pages 1744–1757, 2012.

[Lin *et al.* 2014] T. Lin, Ma S. and Zhang S. *On the convergence rate of multi-block ADMM*. Optimization and Control, 2014.

[Lowe 2004] D.G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*. International Journal of Computer Vision, vol. 60, no. 2, pages 91–110, 2004.

[M. *et al.* 2011] Werlberger M., Thomas Pock, Markus Unger and Horst Bischof. *Optical flow guided TV-L1 video interpolation and restoration*. In Energy Minimization Methods in Computer Vision and Pattern Recognition, 2011.

[Nagel 1990] H. Nagel. *Extending the 'Oriented Smoothness Constraint' into the Temporal Domain and the Estimation of Derivatives of Optical Flow*. In In Proceedings of ECCV '90, 1990.

[Papenberg *et al.* 2006] N. Papenberg, Andrés Bruhn, Thomas Brox, Stephan Didas and Joachim Weickert. *Highly Accurate Optic Flow Computation with Theoretically Justified Warping*. International Journal of Computer Vision, vol. 67, pages 141–158, 2006.

[Randriantsoa & Berthoumieu 2000] A. Randriantsoa and Y. Berthoumieu. *Optical flow estimation using forward-backward constraint equation*. In ICIP, 2000.

[Salgado & Sánchez 2007] A. Salgado and Javier Sánchez. *Temporal Constraints in Large Optical Flow Estimation*. In Computer Aided Systems Theory-EUROCAST '07, 2007.

[Shiratori *et al.* 2006] T. Shiratori, Yasuyuki Matsushita, Sing Bing Kang and Xiaoou Tang. *Video Completion by Motion Field Transfer*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2006.

[Shubhangi 2012] G. Shubhangi. *An Algorithm of TV for Image Inpainting*. International Journal on Computer and Electronics Research, vol. 1, no. 3, 2012.

[Smith *et al.* 2006] A. Smith, Wall MB, Williams AL and Singh KD. *Sensitivity to optic flow in human cortical areas MT and MST.* European Journal of Neuroscience, vol. 23, pages 561–569, 2006.

[Sun *et al.* 2010] D. Sun, S. Roth and M. Black. *Secrets of Optical Flow Estimation and Their Principles.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010.

[Tang *et al.* 2011] N.C. Tang, Chiou-Ting Hsu, Chih-Wen Su and T.K. Shih. *Video Inpainting on Digitized Vintage Films via Maintaining Spatio-Temporal Consinuity.* IEEE Transaction on Multimedia, vol. 13, no. 4, pages 602–614, 2011.

[Volz *et al.* 2011] S. Volz, A. Bruhn, L. Valgaerts and H. Zimmer. *Modeling Temporal Coherence for Optical Flow.* In Proceedings of ICCV '11, 2011.

[Wedel & T. 2008] A. Wedel and Pock T. *An improved algorithm for TV-L1 optical flow.* In In Proc. of Dagstuhl Motion Workshop, 2008.

[Werlberger *et al.* 2009] Manuel Werlberger, Werner Trobin, Thomas Pock, Andreas Wedel, Daniel Cremers and Horst Bischof. *Anisotropic Huber-L1 Optical Flow.* In Proceedings of the British Machine Vision Conference (BMVC), London, UK, September 2009. to appear.

[Wexler & Irani 2007] Y. Wexler and M. Irani. *Space-time Completion of Video.* IEEE transaction on Pattern Analysis and Machine Intelligence, vol. 29, no. 3, pages 463–476, 2007.

[Wulff *et al.* 2012] J. Wulff, others*et al. Wulff, J. and Butler, D. J. and Stanley, G. B. and Black, M. J.* In ECCV, 2012.

[Zhang *et al.* 2010] Z. Zhang, Arvind Ganesh, Xiao Liang and Yi Ma. *TILT: Transform Invariant Low-rank Textures.* In In Proceedings of ACCV '10, 2010.

[Zimmer *et al.* 2011] Henning Zimmer, Andres Bruhn and Joachim Weickert. *Optic Flow in Harmony.* International Journal of Computer Vision, vol. 93, no. 3, pages 368–388, 2011.