DEPTH MAP RECOVERY FROM VIDEOS

BY

JIQIN WANG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Adviser:

Professor David Forsyth

ABSTRACT

The depth map of a video is a very important piece of information. Recovering the depth map of a video expands a 2D video into its 3rd dimension, and creates new possibilities, such as, object insertion, conversion to 3D, shallow depth of field simulation.

In this work, we introduce our approach of recovering depth maps from a video sequence with a moving camera and moving objects.

Our approach isolates moving objects of each frame and estimates the depth of the scene and the moving objects separately. It takes advantage of the fact that the surfaces that belong to the same object share similar optical flow angles, and have smooth optical flow angle gradients, that can be exploited to recover object boundaries, thereby isolating moving objects from the static part of the scene.

It recovers the relative depth of the static part of the scene by calculating the likelihood of a pixel belonging to the farthest background using the magnitude of the optical flow and recovered 3D points. It then estimates the depth of moving objects by finding a statistically most likely actual size of the object and converting the actual size to its actual depth. Finally, we reinsert the estimated depth moving object into the estimated depth of the rest of the scene.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Dataset

For this project we use Sintel, a short computer animated video produced by Ton Roosendaal and the Blender Foundation [1]. Sintel comes with ground truth optical flow and ground truth depth map that allow us to evaluate our results.

The short sequences Sintel provides are challenging. The sample sequences used in the experiments are taken with moving cameras of objects moving at relative high speed.

The following two datasets will be used as examples to illustrate the depth recovery approach. The first sequence shown in figure 1.1, the temple sequence, will be to illustrate moving object depth handling. It has a large range of depth, and two moving objects with different speeds. The other sequence shown in figure 1.2, the alley sequence, will be used to illustrate scene depth estimate. The background depth varies as the girl runs quickly across the alley.

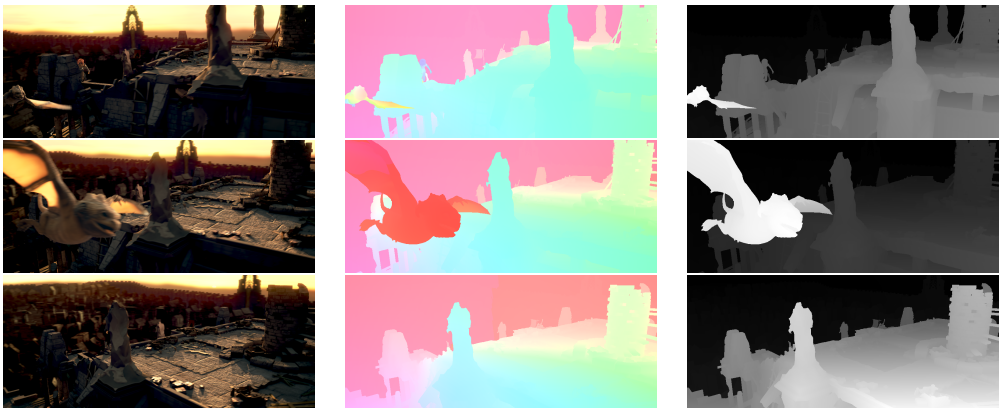


Figure 1.1: **Left:** Image Frame, **Center:** Ground Truth Optical Flow (encoded such that its hue shows the direction of the flow, saturation shows the magnitude of the flow) and **Right:** Ground Truth Depth From The Temple Sequence (darker values indicates greater depth)

Figure 1.2: **Left:** Image Frame, **Center:** Ground Truth Optical Flow (encoded such that its hue shows the direction of the flow, saturation shows the magnitude of the flow) and **Right:** Ground Truth Depth From The Alley Sequence (darker values indicates greater depth)

1.2 Inspiration and Existing Softwares Results

The inspiration for this method comes from two existing approaches. One is ACTS 2.0 from Guofeng. Zhang [2], the other is depth transfer from Kevin Karsch[3].

1.2.1 ACTS

ACTS[2] generates outstanding depth results that are spatially and temporally consistent and accurate when the scene is static without moving objects. It aims to achieve photo consistency and geometric coherence across frames. While the goal guarantees structural consistency of the static part of a scene, it does not take into account moving objects in the scenes. Moving objects in each frames are very likely to be considered outliers, thus not rendered correctly into the depth map of the scenes. See figure 1.3

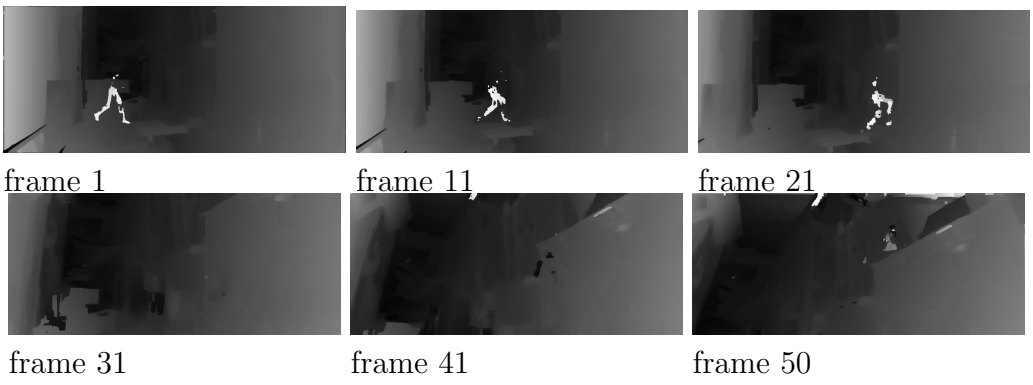
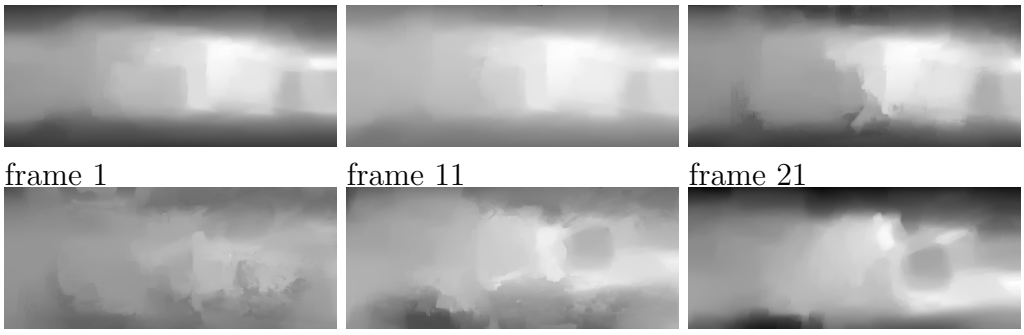


Figure 1.3: Acts Results
The structure of the alley is clearly rendered with decent amount of details, however, the presence of the girl is not handled as well as the static part of the scene

1.2.2 Depth Transfer

Depth transfer[3] does not require a moving camera and predicts the depth of moving objects in addition to the scene, however it lacks details compared to ACTS. See figure, 1.4



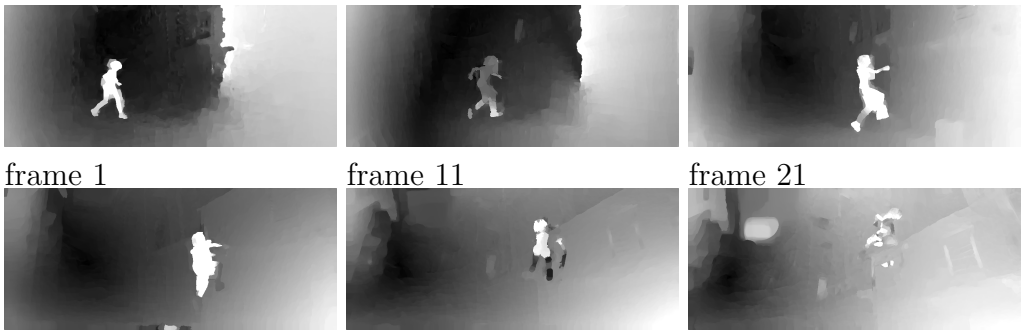
frame 1 frame 11 frame 21

frame 31 frame 41 frame 50

Figure 1.4: Depth Transfer Results

1.2.3 Our Approach

Inspiration for our method comes from trying to fuse the results of ACTS and Depth transfer together. Our approach seeks to separate moving objects from the scene and estimate their depth separately. However, there is still one last step missing to fuse the two parts together, therefore, the results shown here do not have correct moving object depth inserted. See figure, 1.5



frame 1 frame 11 frame 21

frame 31 frame 41 frame 50

Figure 1.5: Our Results

CHAPTER 2

APPROACH SUMMARY

2.1 General Idea

The idea is to utilize the motion of the camera or the scene, and therefore our approach is not applicable to videos of moving objects with static scene and fixed camera.
We split the depth map reconstruction into two parts: moving object depth recovery and scene depth estimation.

Scene Depth Recovery

We estimate the scene depth map by tracking and analyzing the motion of the objects in a video with two basic information: 1.) tracked points; 2.) optical flow. Tracking feature points and recovering camera matrix relates 2D image coordinates with 3D world coordinates, whose depth can be easily extracted.
Optical flow gives the angle and magnitude of motion of each pixel. And our approach is based on two basic assumptions of optical flow.

- Ideally, static rigid bodies that are closer to the camera appear to move faster than those that are farther from the camera.
  Figure 2.1 is a heat map of the magnitude of the ground truth optical flow of the first frame (the warmer the color the faster the movement). As shown in the image, the pillar that is closest to the camera is colored in dark red (largest movement), while the pillars that are on the other side of the temple are colored in dark blue (smallest movement).

- Ideally, the surfaces of a static rigid body tends to have similar angles of movement and smooth angles gradient.That is to say, for a rigid

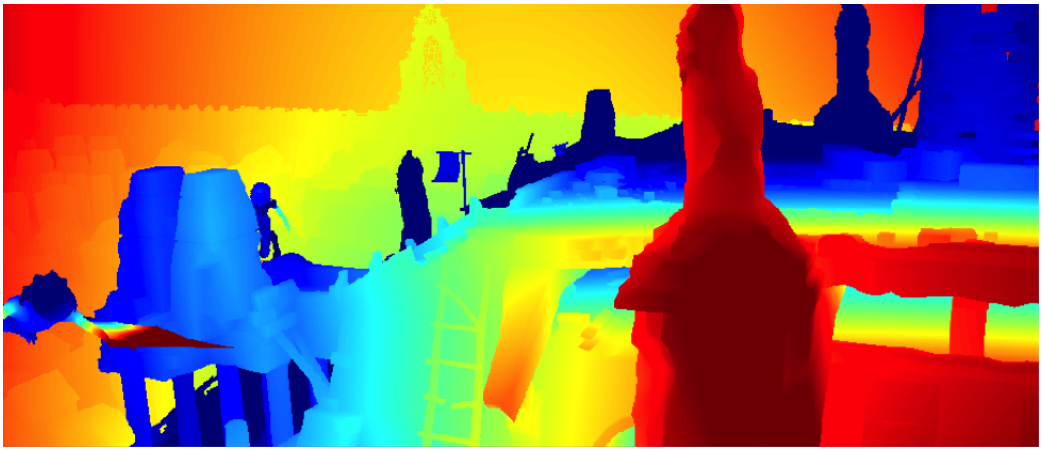


Figure 2.1: Magnitude of Optical Flow Heat Map
Warmer colors indicates larger magnitude.

body that is static with respect to the scene, all its surfaces will appear to moves at similar speed and on the same surface the speed varies smoothly.

Figure 2.2 is the magnitude of the gradient of angles of movements, which shows clear contours of each different rigid body.

Figure 2.2: Thresholded Magnitude of Gradient of Optical Flow Angles

Moving Object Depth Estimation

The perceived speed of a moving object is inversely proportional to its depth, the farther it is, the slower it appears to move. To estimate the depth of moving objects, we choose a statistical approach. The approach assumes that the

speeds at which objects move and the size of the objects in videos follows certain underlying distributions. For example, the size of the moving objects in a video follows a normal distribution with average size of 1.5 meters.

2.2 Steps Summary

Here is a brief overview of the major steps of the depth recovery approach. More details can be found in Chapters 3, 4 and 5.

1. Compute forward and backward[1] optical flow of the sequence
   Track feature points, estimate camera matrices, reconstruct 3D points.

2. Recover relative depth of the static part of each frame with optical flow and tracked points

3. Perform motion segment on each frame, identify moving objects

4. Estimate depth of moving objects in each frame

5. Put moving object depth into scene depth

2.2.1 Optical Flow Calculation and Point Tracking

Optical flow calculation is the first and most important step in the depth map estimation approach. Our approach is very sensitive to the quality of the optical flow. It affects the scene depth recovery, motion segmentation and in turn affects moving object depth estimation.
There are many existing softwares that generate results with good accuracy. In this work, we used the software provided by D. Sun [4].
Point tracking provides absolute depth that can aid scene depth recovery,

[1] Optical flow from $frame_i$ to $frame_{i-1}$

and motion segmentation. It is not as essential as optical flow, as we do not need absolute depth. In this work, we used the point feature tracking in [2].

2.2.2 Scene Depth Recovery

According to assumption one (2.1), the magnitude of optical flow of a video gives abundant information for scene depth recovery as long as the camera is not fixed.
We use optical flow as well as tracked 3D points, if any, to generate a relative depth representation of each frame. In this step, we process each frame as a whole without worrying about moving objects.
Because of assumption two (2.1), static rigid objects in a scene all move with similar angles and the angles and magnitude of their movement has smooth gradient. In contrary, a moving object object, in other words it is not relatively static, will appear to move in angles that do not accord with the rest of the scene. Thereby, this step also gives important information for motion segmentation.

2.2.3 Motion Segmentation

Motion segmentation depends mainly on assumption two 2.1, which says static rigid objects in a scene moves in similar angles and has smooth motion gradient. Since the surfaces of the same object moves at similar angle, we can just threshold on the angles and cluster objects that have the same movement and create a segmentation for each frame.
Also it seems that it is hard to for feature points to cling on moving object, which means, most of the time, moving objects usually do not have feature points that are consistent through out the video. This fact can then be exploited to sift out the segments that belong to static objects, so we are left with only segments that belong to moving objects.

2.2.4 Moving Object Depth Estimation

After motion segmentation we have a mask of moving objects for each frame. Then we can perform depth estimation using the approximate 2d speed of

the object and the size estimate. More detail will be explained in chapter 5.

CHAPTER 3

SCENE DEPTH ESTIMATION

3.1 Main Idea

Our scene depth estimation has drawn inspiration from GrabCut[5] segmentation. In GrabCut segmentation, each pixel is given a data cost that is the likelihood of assigning that pixel to the background or foreground.[6]
Scene depth estimation uses the same idea, instead of using RGB values for data cost computation, we use the optical flow. In our method, a relative depth of a pixel is the data cost to assign that pixel to the background[5]. However, here by background, we are referring to the farthest part in the scene.
Farthest background region is determined in two different ways depending on whether reconstructed 3D points are available and reliable. The methods are explained in detail in the following sections.
Here we take the alley sequence as an example to illustrate these methods. The input of this step is original frame, optical flow[1] and 3D tracked points, as shown in figure 3.1.[2]

(a) Original Frame

(b) Optical Flow
Hue encodes direction of the flow, saturation encodes the magnitude of the flow

(c) 3D Points
Warmer color means larger depth

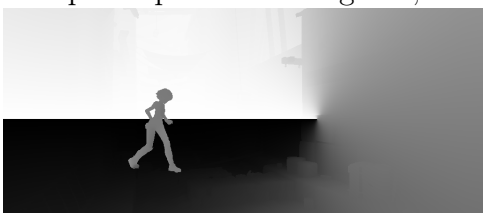
Figure 3.1: Inputs to Scene Depth Estimation

[1]To better show the ideal effects of each step, in this example we use ground truth optical flow as input.
[2]Points shown here are color-coded, warmer colors indicates larger depth

3.2 3D Point Guided Method

In this method we use the reconstructed 3D points to show where the farthest part on the image is.

1. Compute optical flow angle Θ, and magnitude R



Optical Flow Angle Θ

Optical Flow Magnitude R

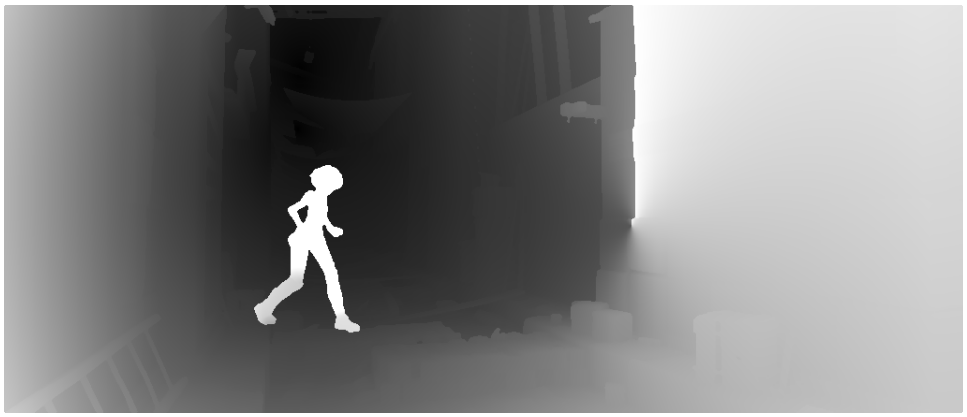
2. Sort the reconstructed 3D points by Z(depth), choose the top 5% farthest points, $\{(X, Y, Z)\}$ and calculate average R values at these points.

$$\bar{r} = \frac{1}{|X|}\Sigma_{(x,y)\in(X,Y,Z)}R(x, y) \tag{3.1}$$

Original Frame with Top 5% Farthest Points

3. Calculate data cost for each pixel with R.

$$\Sigma(((R - \bar{r}) \cdot cov(R(X, Y))^{-1}) \cdot (R - \bar{r})/2) \tag{3.2}$$

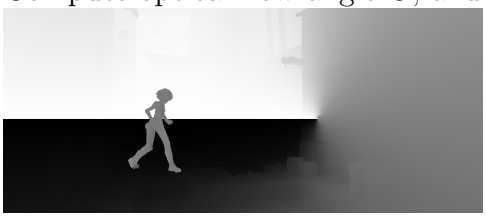


Final Scene Depth Estimation
[6] Darker pixels have larger depth

3.3 Heuristic Based Method

This method does not use 3D points to deduce the farthest region in the image, instead it uses a heuristic function to identify the farthest background region. The heuristic is based on the assumption that the image segment that correspond to the farthest background region has the smallest average optical flow magnitude.
Compared to previous method, this method lifts dependency on quality of predicted 3D coordinates, but is less robust than 3D Point Guided Method.

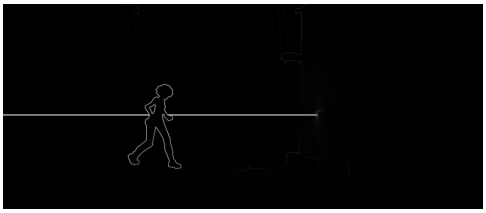
1. Compute optical flow angle Θ, and magnitude R



Optical Flow Angle Θ

Optical Flow Magnitude R

2. Compute gradient of Θ in polar coordinates $\frac{\delta\Theta}{\delta r}$, and watershed segment $\frac{\delta\Theta}{\delta r}$



Magnitude of Gradient of Flow Angle $\frac{\delta\Theta}{\delta r}$

Watershed Segmentation

3. Compute for each segment ω, the average R in each region divided by the area of the region $\frac{\bar{r}_\omega}{area(\omega)}$, the segment that has the farthest background region is given by

$$\omega_{back} = argmin_\omega \frac{\bar{r}_\omega}{area(\omega)} \tag{3.3}$$

4. Calculate average optical flow magnitude $\bar{r}$ of region $\omega back$ as $\bar{r}_{\omega back} = \frac{1}{|X|}\Sigma_{\Omega\in(\omega_{back})}R(x,y)$

5. Calculate data cost for each pixel with R[6].

$$\Sigma(((R-\bar{r}_{\omega back})\cdot cov(R(\omega_{back}))^{-1})\cdot(R-\bar{r}_{\omega back})/2) \tag{3.4}$$

Final Scene Depth Estimation

Darker pixels have larger depth

3.4 Results

Because scene depth is computed directly from optical flow magnitude and background segmentation, background identification also utilizes optical flow angle for segmentation, the quality of this step heavily relies on the accuracy of optical flow magnitude.

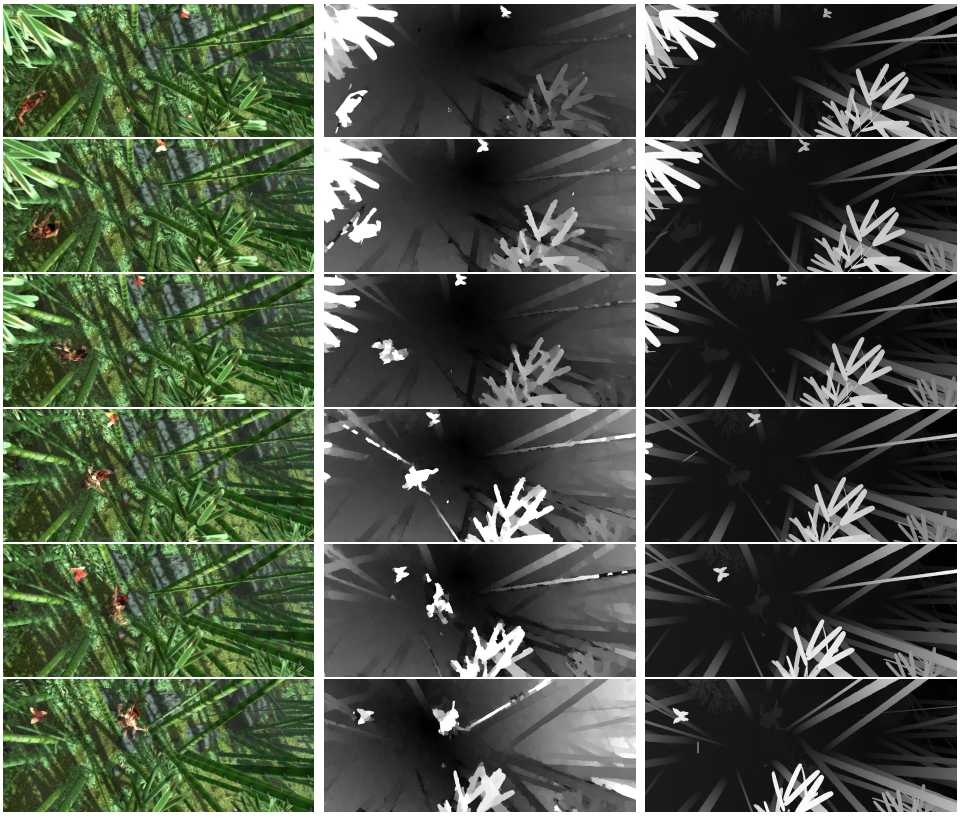
The scene depth predicted from estimated flow is significantly noisier than those predicted from ground truth flow. For example, the last example in the second column has a very light spot in the sky.

Ignore the depth of the person in the results as it will be handled separately in Chapter 5.

Figure 3.2: Alley Scene Depth Estimation Results

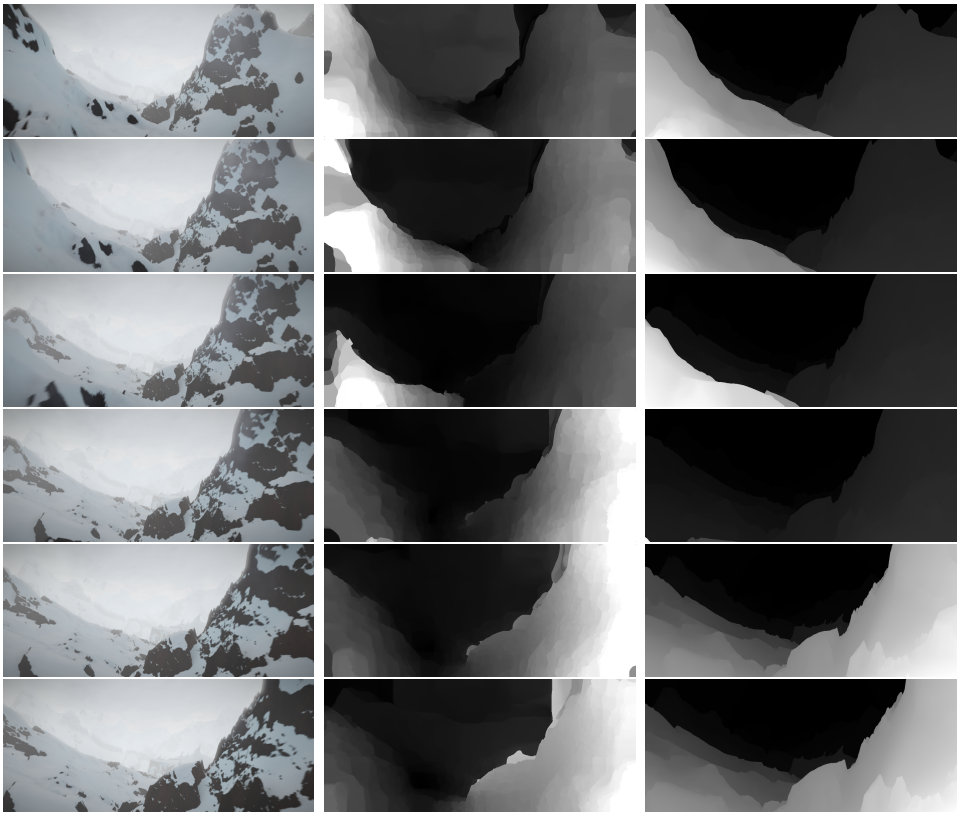
Original Frame | Predicted Depth | Ground Truth Depth

Figure 3.3: Bamboo Scene Depth Estimation Results



Original Frame | Predicted Depth | Ground Truth Depth

Figure 3.4: Mountain Scene Depth Estimation Results



Original Frame | Predicted Depth | Ground Truth Depth

CHAPTER 4

MOTION SEGMENTATION

4.1 Main Idea

For motion segmentation we fully utilize assumption two (2.1), that the angle of optical flow does not vary much across the same object to separate different surfaces. Clustering or watershed segment on the gradient of the angle of the optical flow results in segmented surfaces.
Then we took advantage of the fact that, tracked feature points used by ACTS[2] are fairly dense and cannot be on objects that are moving. Based on that, we can extract background by eliminating segments that have feature points on them.
However, predicted optical flow is usually noisy and can lead to unpredictable results, to make the segmentation more robust we also use the previous result to guide subsequent results and then use subsequent segmented result to clean up previous results, we call this procedure "negotiation".

4.1.1 Negotiation Idea

Optical flow gives the magnitude and direction of the movement of the pixel, therefore we should be able to predict segmentation of frame $i+1$, seg_{i+1} by adding optical flow of frame i, $flow_i$ to segmented result of the previous frame seg_i. Similarity with backward flow, we should be able to predict previous segmentation with current frame by adding backward flow $flow_back_i$ to current segmentation seg_i
Since both results are from estimated flow, and predicted segmentation, we will need to determine, for the pixels that the two results do not agree, which segment should they belong.

To illustrate this idea, we will use estimated flow as input, as using ground truth flow can give cleanly segmented moving objects with just the segmentation step

4.2 Algorithm

1. Compute optical flow angle Θ, and magnitude R

Optical Flow Angle Θ

Optical Flow Magnitude R

2. Compute gradient of Θ, and obtain watershed segmentation

Gradient of Optical Flow Angle Θ

Segmented According to Optical Flow Angle Θ

3. Clean up initial segmentation by merging small segments to the a segment that is larger and has similar optical flow angle.

Merged Small Segments
Compared with the segmentation in previous step, the small segments on the ground have disappeared.

4. Background elimination

   (a) Let k be the number of minimum feature points to be on a segment for the segment to be considered background segment.

   (b) Eliminate background by eliminating the segments that have more than k feature points on them.

   (c) If doing so end up with less than one other segment on the frame, $k = k + 1$ go to step (b)

Segmentation with 2D Feature Point Overlayed

Segments with > 2 Points Are Eliminated

5. Negotiate between frames

   (a) If current frame i is not the first frame, negotiate segmentation of current frame, seg_i by adding $flow_{i-1}$ to seg_{i-1}

   (b) If current frame is the second frame, negotiate seg_{i-1} by adding backward flow of current frame, $flow_back_{i+1}$ to current segmentation seg_i

4.3 Negotiation Algorithm

Optical flow gives the magnitude and direction of the movement of the pixel, therefore we should be able to predict $frame_i$ with $frame_{i-1}$ and its optical flow $flow_{i-1}$.
This procedure takes optical flow of the previous frame $flow_{i-1}$ previous segmentation seg_{i-1}, current segmentation seg_i and current frame image $frame_i$(can be gray scale or colored) as inputs.

4.3.1 Notations

A list of notations used in this section

- seg_i: Computed segmentation of frame i
- s: Single Segment
- S: Set of segments
- $boundary(s_i, s_j)$: boundary pixels in side segment i, neighboring segment j
- $predSeg_i$: Predicted segmentation of frame i, can be from frame i-1 or frame i+1
- $flow_i$: Optical flow of frame i
- $frame_i(s)$: pixel values inside segment s of frame i

4.3.2 Algorithm

I Predict current frame segmentation by adding previous flow to previous segmentation $predSeg_i = seg_{i-1} + flow_{i-1}$

II Clean up predicted segmentation $predSeg_i$ by filling any holes it has.

III Calculate the difference between predicted segmentation and current segmentation $diffSeg = predSeg_i \oplus seg_i$

IV Relabel each connected components in the difference as a different segment $diffSeg = Label(diffSeg)$

V For each segment s in $diffSeg$

(a) If its size is larger than $reSegThreshold$

i Further segment into k pieces, where $k = area(s)/reSegThreshold$

ii Run KMeans on s with input frame $frame_i(s)$

iii Merge the newly created segments in to $diffSeg$

VI Sort the segment in $diffSeg$ with ascending area size

VII For each segment s in $diffSeg$ (area size from small to large)

(a) Find all its neighboring segments $S_{neighbor}$

(b) For each segment s_n in $S_{neighbor}$

i Find the pixels in s_n that are on the boundary of s, compute average

$$\overline{val_{s_n}} = avg(boundary(s_n, s)) \tag{4.1}$$

ii Find the pixels in s that are on the boundary of s_n, compute average

$$\overline{val_s} = avg(boundary(s, s_n)) \tag{4.2}$$

iii Find the euclidean distance between s and s_n

$$diffval = \sqrt{(\overline{val_s} - \overline{val_{s_n}})^2} \tag{4.3}$$

(c) Merge current segment s into the neighbor with minimum $diffval$

4.3.3 Sample Results From Each Step

Initial Segmentation of Current Frame

Segmentation of Previous Frame

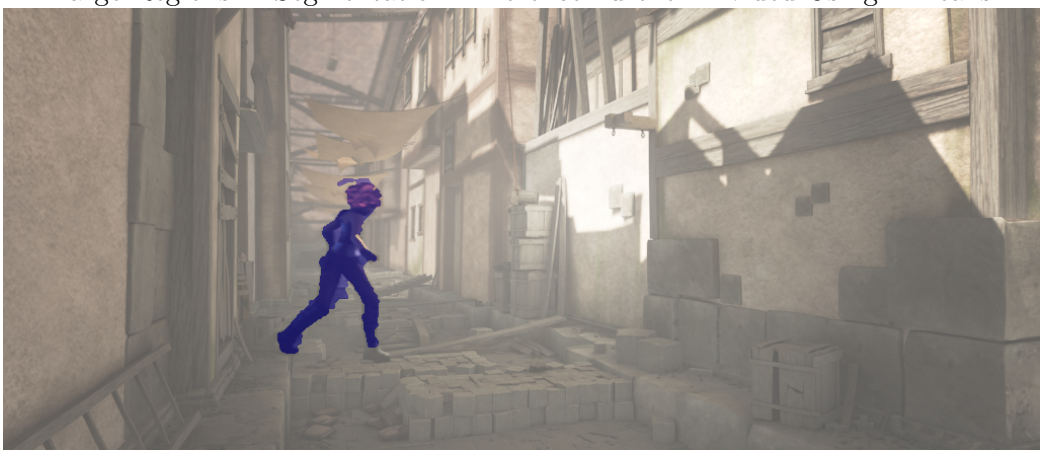
Predicted Current Segmentation
by adding the optical flow of previous frame to previous image frame $seg_{i-1} + flow_{i-1}$

Predicted Segmentation with Cracks Filled

Predicted and Current Segmentation Difference

Segmentation Difference with Connected Component Labeled

Large Regions in Segmentation Difference Further Divided Using KMeans



Segmentation Difference Merged Into Either Background or Foreground According to RGB Values

Negotiated Difference

Purple segments are the part that used to belong to the foreground but marked as background after negotiation procedure.

Figure 4.1: Per Step Negotiation Results

4.4 Results

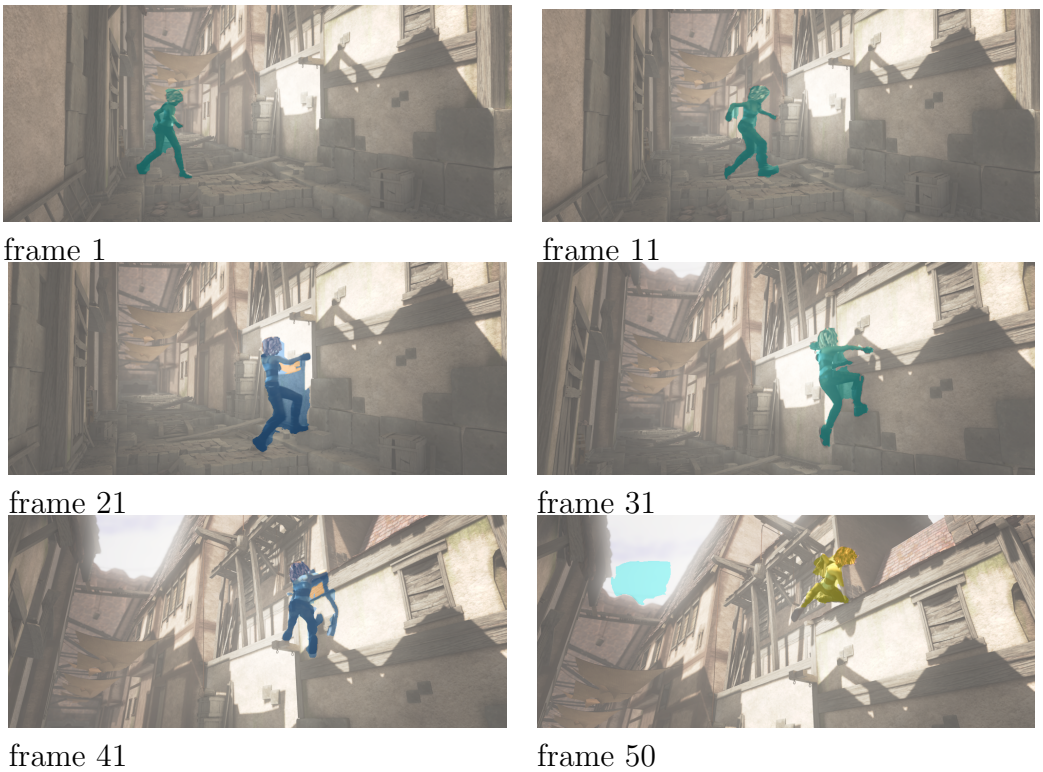


Figure 4.2: Alley Motion Segmentation Results

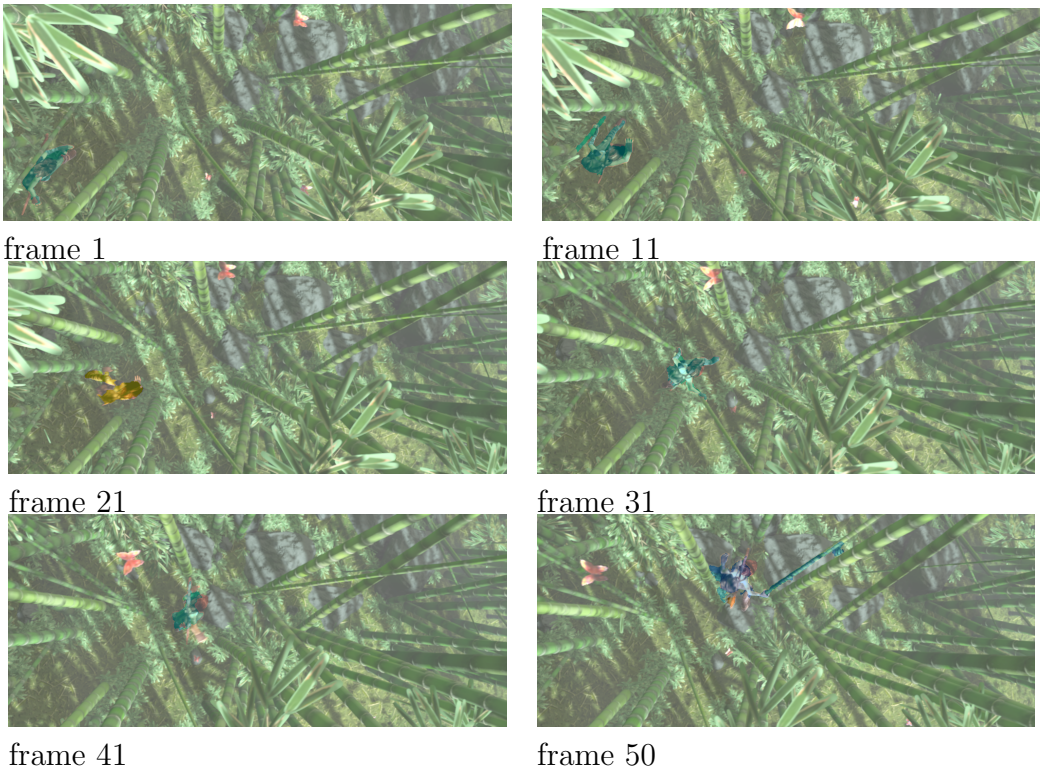


Figure 4.3: Bamboo Motion Segmentation Results

CHAPTER 5

MOVING OBJECT DEPTH ESTIMATION

5.1 Main Idea

We assume that in natural world the size of objects that can move, and the speed at which they move follow certain underlying distribution. Let D be under distribution $\Pr_D$, V be under distribution $\Pr_V$
Let an sphere with diameter D distance Z from a camera with focal length f be projected on a canvas, the projected image will be a circle with diameter $d = \frac{D \cdot f}{Z}$. Similarity, a moving sphere with speed V when taken by a video camera, the sphere will have a 2D speed of $v = \frac{V \cdot f}{Z}$.

$$d = \frac{D \cdot f}{Z} \tag{5.1}$$

$$v = \frac{V \cdot f}{Z} \tag{5.2}$$

$$\Rightarrow \frac{D \cdot f}{d} = \frac{V \cdot f}{v} \tag{5.3}$$

$$\Rightarrow V = \frac{D \cdot v}{d} \tag{5.4}$$

Therefore, the problem of moving object depth estimation can be reduced to finding the original size D of the object. And the problem of can be stated as maximize the probability of size D of the object given the size of the object on the image and the 2D speed of the object at which it moves.

$$\Pr(D|d, v) = \frac{\Pr_D(D) \cdot \Pr_V(\frac{v}{d} \cdot D)}{\int_0^\infty (\Pr_D(D) \cdot \Pr_V(\frac{v}{d} \cdot D))dD} \tag{5.5}$$

Figure 5.1: Objective Function

When applied to a video sequence, in analogous to the diameter of a circle, in 2D videos we use equivalent diameter as d, it is computed as:

$$d = \sqrt{\frac{4 \cdot area(s)}{\pi}} \tag{5.6}$$

5.2 Background Models

Here we model the size D of a moving object as a Gamma distribution (the unit of D is in meters) which means the average equivalent diameter of a moving object is 1.31 meters.

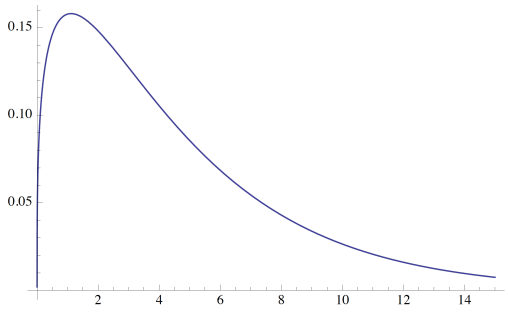
$$\Gamma(a) = \frac{x^{\alpha-1}}{\Gamma(\alpha)\beta^{\alpha}} exp(-\frac{\beta}{x}) \tag{5.7}$$

$$with\ \alpha = 1.31,\ \beta = 3.6 \tag{5.8}$$

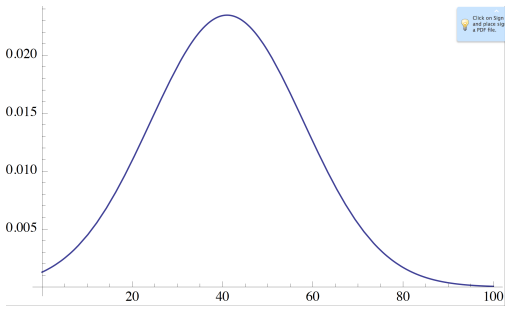
We model the speed at which it moves as a Normal distribution. The unit of V is pixels per frame, the amount of displacement between two frames in pixels. Because our objective function variable D is in meters, we need to convert V into meters as well. We will multiply D by 1000 pixels per millimeter $1000 * (pixels/mm)$ to convert D to pixels.

$$V \sim \mathcal{N}(\mu, \sigma^2) \tag{5.9}$$

$$with\ \mu = 41,\ \sigma = 17 \tag{5.10}$$



Distribution of Moving Object Size in Meters



Distribution of Moving Object Speed in Pixels Per Frame

Figure 5.2: Background Model Plots

5.3 Depth Estimation

5.3.1 Inputs

We take the optical flow for each frame, motion segmentation results from each frame and the focal length of this video sequence as inputs.
Parameter pixel per millimeter $ppmm$, here we use $ppmm = 1200$

5.3.2 Algorithm

I For each frame i

(a) For each moving segment s_{ik}

i. Compute optical flow magnitude R_i

ii. Calculate equivalent diameter d_{ik} of the moving segment in this frame

iii. Get the mean 2D speed $v_{ik} = avg(R_i(s_{ik}))$ of this segment

iv. $ratio = \frac{1000 * ppmm}{f}$
Find D that maximizes objective function **??** $D_{ik} = argmax\big(\Pr_D(D) \cdot \Pr_V(\frac{ratio \cdot v}{d} \cdot D)\big)$

II For each object k

(a) Smooth D_k with moving average smoothing with window size 5

III For each frame estimate depth by

(a) $Z_{i_k} = \frac{1000 \cdot D \cdot f \cdot ppmm}{d_i}$

5.4 Inputs and Results

5.4.1 Inputs

Here we use frame one through fifteen as temple sequence as an example, as both the girl and the dragon exist in the frames.5.3
In the original video, the dragon flies quickly towards the camera and then

goes out of the frame. The 2D speed v and 2D size d of the dragon increases rapidly during frame 1-15.

Original Frames

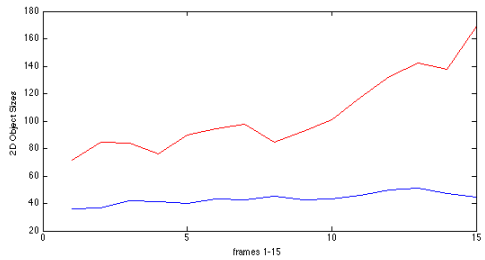
Frame 1 Frame 3 Frame 6

Frame 9 Frame 12 Frame 15

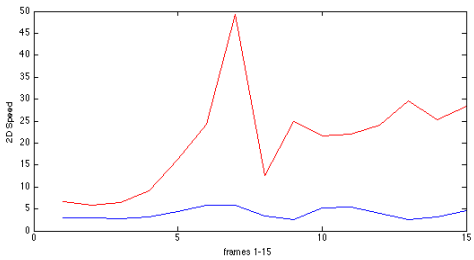
Figure 5.3: Original Frames

2D Sizes and Velocities

See figure, 5.4



Segment Equivalent Diameter Used as 2D Size d
Size of dragon in red and size of the human in blue

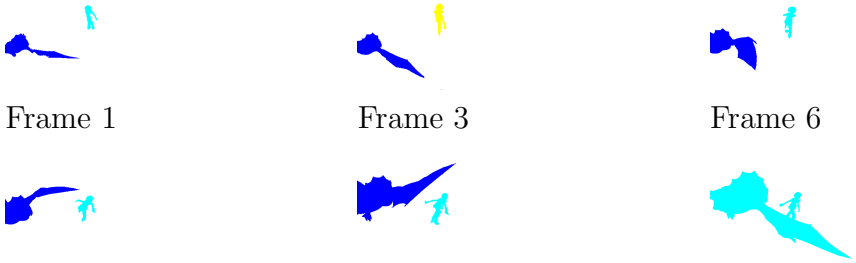


2D Median Optical Flow Used as 2D Velocity v
Velocity of dragon in red and velocity of the human in blue

Figure 5.4: 2D Sizes and Velocities

Segmented Frames

Manually segmented results of the dragon and the human. 5.5



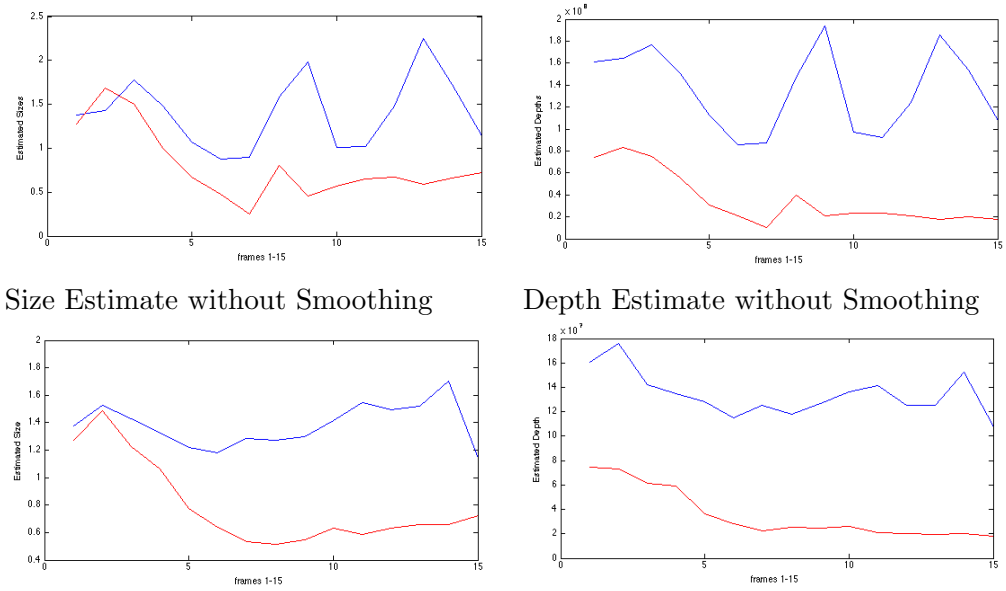
Frame 1 Frame 3 Frame 6

Frame 9 Frame 12 Frame 15

Figure 5.5: Frames with Dragon and the Person Manually Segmented
Label color is chosen randomly.

5.4.2 Estimated Depth Results

See figure 5.6



Size Estimate without Smoothing Depth Estimate without Smoothing

Size Estimate with Smoothing Depth Estimate with Smoothing

Figure 5.6: Estimates with Smoothing
Estimates of the dragon in red and the human in blue.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Future Work

We were not able to finish inserting the estimated moving object depth back into the estimated scene depth map, and there are also many improvements we need to make to the existing scene depth generation and motion segmentation approach.

6.1.1 Moving Object Insertion

When there are more than two object exist in the scene, we are not yet able to track them across frames. But this should not be too hard to do by tracking their respective motion in the scenes.

6.1.2 Scene Depth Temporal Consistency

Currently, scene depth of each frame is computed separately, and still lacks temporal consistency, and is sensitive to noise that appears in estimated flow and 3D points.

6.2 Conclusion

Accurate depth recovery from video is a challenging problem. Even though the assumptions we make in our approach holds true in general, approximated optical flow and estimated 3D points do not always behave as the assumptions expect. Therefore, we will need extra steps, such as negotiation, to verify

and correct possible error. More work should be done to make scene depth and object more consistent to improve overall accuracy in depth recovery.

REFERENCES

[1] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.

[2] T.-T. W. Guofeng Zhang, Jiaya Jia and H. Bao, "Consistent depth maps recovery from a video sequence," vol. 31, no. 6, pp. 974–988, jun 2009.

[3] K. Karsch, C. Liu, and S. B. Kang, "Depthtransfer: Depth extraction from video using non-parametric sampling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2014.

[4] S. R. Deqing Sun and M. J. Black, "A quantitative analysis of current practices in optical flow estimation and the principles behind them," *International Journal of Computer Vision*, vol. 106, pp. 115–137, Jan. 2014.

[5] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut -interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (SIGGRAPH)*, August 2004. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=67890

[6] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for markov random fields," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 16–29.