Fun with ChromaKey and MATLAB

Automatic person segmentation in webcam video for ChromaKey application

Nkiruka Chuka-Obah
Electrical Engineering
Stanford University
Stanford, California
nchukaobah@gmail.com

***Abstract*— The primary goal of this project is to implement a MATLAB toolbox for video conferencing application of Chroma Key for single user webcam. This toolbox combines background subtraction methods [3][4] with motion segmentation via optical flow [7][5] to segment out the user in a foreground image and superimpose this foreground unto a user-desired background image. In this report, the overall algorithm for segmentation is described, with a description of the background subtraction techniques implemented, as well as the optical flow algorithms used. Finally, the algorithm is tested and results and improvements are described.**

Keywords-component; background subtration; motion detection;optical flow; chroma key; MATLAB;

I. INTRODUCTION

Chroma Key is a popular video editing technique that involves the superposition of the foreground in a video sequence onto a desired background. It has applications in the generation of special effects in television, movie production, video conferencing as well as in photo editing. However, Chroma Key has several drawbacks. Images to be segmented must have a solid color background (usually blue or green) that is significantly different from that of the foreground of interest, making it easy to replace this solid color with a different background. In addition, significant user interaction is often required to segment the foreground onto the desired background image. This project proposes an automatic, foreground segmentation algorithm that removes the need for a solid color background and user intervention. In addition, it describes the accompanying MATLAB toolbox for the algorithm. The application of the toolbox is for webcam video conferencing with a single user. The algorithm in the toolbox uses background subtraction, and motion detection via optical flow techniques to detect and segment the foreground from the scene. This paper is organized as follows: Section II gives an overview of the algorithm as well as the setup of the environment. Section III describes the background subtraction techniques used in the algorithm. Section IV describes the optical flow techniques used in the algorithm. Section V lists and discusses the results obtained as well as improvements to the toolbox, with section VI giving conclusions of the paper.

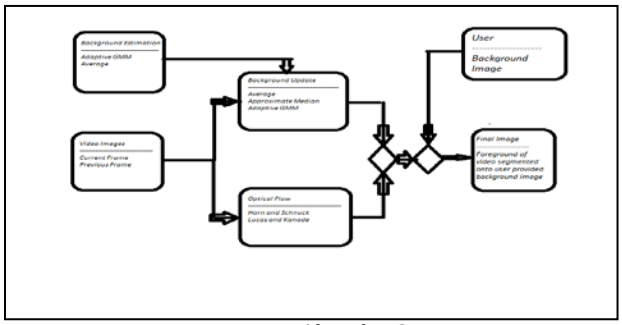

Figure 1. Algorithm Overview

II. OVERVIEW

A. Algorithm Overview

The algorithm overview is shown in Fig. 1. The steps for the algorithm are shown below:

- Train the background using *N* frames of video images (Fig. 3). The number N (default 50 frames) and the background subtraction method for training are chosen by the user. While frames of only the background provide better results in segmentation, it is not necessary for the algorithm. The result of training is the background image B_{t-1}.
- The real-time phase of the algorithm, video conferencing begins. A pair of frames, taken every 5^{th} frame from the video stream, is passed to the optical flow method (Fig. 4) chosen by the user. The resulting foreground mask F_t is produced.
- The background image B_{t-1} is updated with the current frame based on the user chosen background subtraction method. The result of the update is B_t .
- The updated background is subtracted from the current image producing a background mask BM_t that is combined with the foreground mask of optical flow F_t, producing the final foreground mask FM_t .
- The user chosen background *BM* is combined with the foreground mask FM_t and the current frame to produce the final Chroma key image.

Each of the steps of the algorithm is displayed in MATLAB figures to the user.

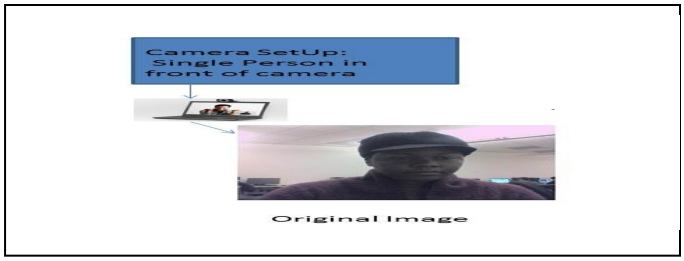

Figure 2. Camera Environment

B. *Camera Environment Setup*

The camera environment is shown in Fig. 2. It comprises a single user, in front of a webcam, with a static background. There are no color requirements for the background of the user.

III. BACKGROUND SUBTRACTION

Background subtraction is a commonly used technique to segment objects in static scenes. A review of the many different methods of forming a model of the background is given in [1]. In this algorithm, background subtraction occurs in two places; during initialization for training the background model, and during run-time for real-time update to the background model. In either stage, the user has the option to choose an implementation of the averaging method [1] or the approximate median method [3] for initialization of the background model.

The initialization of the algorithm involves developing a model of the background. It is shown in Fig. 3. This model is obtained by applying the user chosen subtraction method to N video frames, producing the result B_{t-1} at time t-1.

The update of the background model during run-time, B_t at time t, is performed using either the averaging method [1] updated every 50^{th} frame with the collection of the last 5 frames, or with the approximate median method [3] updated every frame. The result of the update of the background is subtracted from the current frame to produce a difference mask BM_t. This mask multiplies the results of optical flow (during motion) to produce the final mask for the chroma keying step. If no motion occurs, the final mask is simply the difference mask. The background subtraction algorithms are described in the following subsections:

A. *Average of N frames with selectivity*

The average of the previous N frames from start to time t-1 is obtained as described in [1]. The background is obtained as a running average of the previous N frames, with α default of 0.05 using $B_t(x,y) = \alpha * C_t(x,y) + (1- \alpha) * B_{t-1}(x,y)$ if $C_t(x,y)$ background $B_t(x,y) = B_{t-1}(x,y)$ if $C_t(x,y)$ is foreground. C_t is the current frame.

B. *Approximate Median*

The approximate median method is described in [3]. The implementation for the toolbox follows exactly as described in [3]. It is also an improved version of that described in [8]. This method does not need to be trained as it can be learned very quickly during run-time. However, as the running averaging

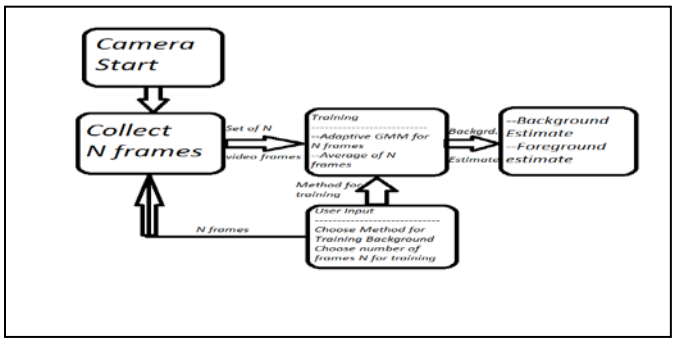

Figure 3. Background Initialization and update

method was provided for the user, the approximate median was added as an option for the user.

IV. OPTICAL FLOW

Optical flow methods are used to detect moving objects in a scene, producing velocity vectors in the x- and y- direction showing motion. While an ideal way to solve the segmentation problem, they must be combined with another method as they only detect when an object is in motion. In the toolbox, the methods described in [5][7] are provided for the user using a MATLAB implementation of [7] and the implementation of [5] provided by Piotr's toolbox [4]. These two methods are chosen as they are widely used in literature, and were easy to incorporate into the algorithm.

The optical flow vectors are calculated at every pixel, rather than at select points as advised by [6]. This was chosen as it was fast enough to not affect real time run.

The optical flow vectors are processed as shown in Fig. 4. The current and previous video frames are passed to the optical flow block. The frames are processed using the user chosen method for optical flow. The velocity vectors, Vx and Vy, produced from the optical flow are filtered with a 7x7 median filter and processed as follows to determine motion:

- The magnitude of the vectors are applied to a threshold (chosen through trial and error) to determine motion. This removes small motion vectors. A binary mask of the flow vectors is obtained from this step.
- The binary mask is morphologically processed using an erosion with a small rectangular structural element, followed by a dilation with a larger rectangular structural element to remove small blobs.
- The result of the morphological processing is passed to a blob analysis block to further remove blobs with area less than a minimum blob area (chosen through trial and error).
- The result of blob analysis is further processed to fill holes in the mask, creating the optical flow mask F_t.

The foreground mask is then combined with the background update to produce the final foreground mask, FM_t for the chroma key mixing step.

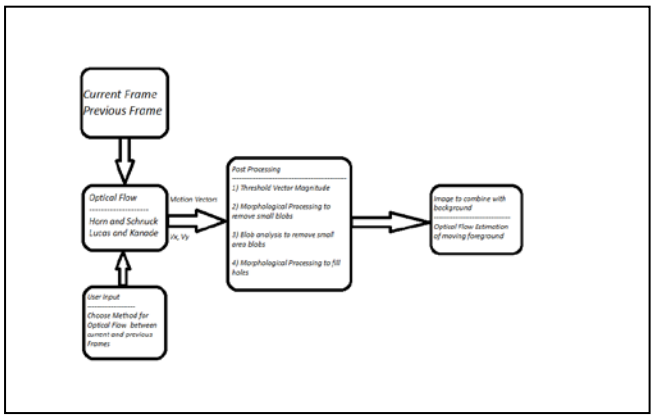

Figure 4. Optical Flow processing

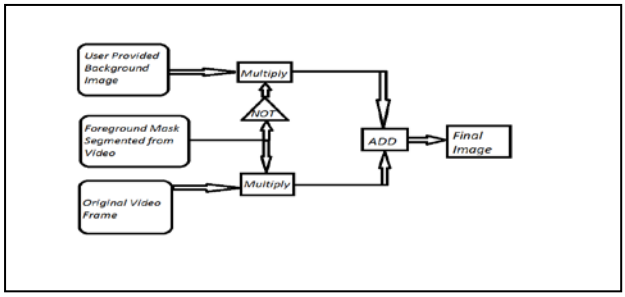

Figure 5. Chroma Key processing

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity "Magnetization", or "Magnetization, M", not just "M". If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write "Magnetization (A/m)" or "Magnetization {A[m(1)]}", not just "A/m". Do not label axes with a ratio of quantities and units. For example, write "Temperature (K)", not "Temperature/K".

V. Chroma Key Combination

The algorithm to combine the segmentation result with the user provided background is shown in Fig. 5. It is a simple multiplication of the algorithm foreground mask, with the video image segmenting out the foreground of interest. This result is added to the product of the inverse of the foreground mask with the provided background.

VI. Results, Discussion And Improvements

The results of segmentation are shown in Fig. 6. The result is shown for an averaging initialization, an approximate median update and the optical flow method of [5]. It was trained using 50 frames, and the result shown is the last image of 100 frames. The results are very poor. Most of the foreground bleeds into the background, creating a halo around the user. This may be due to a poor background model as the background was trained with the user in the scene at all times.

There are several improvements possible for the toolbox. First, a better method of background subtraction is needed. Adaptive mixture models, as described in [2], is the next stage for the project. It is possible for real time [1] and can be incorporated quickly into the algorithm.

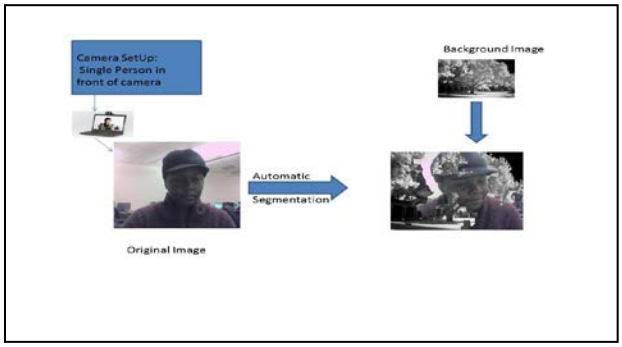

Figure 6. Algorithm Result showing bleeding of background into foreground

Furthermore, a better method to combine the background model with the optical flow mask needs to be developed for real-time. At present, difference mask BM_t multiplies the results of optical flow to produce the final mask for the chroma keying step. If no motion occurs, that is, the sum of the pixels in the optical flow mask F_t is below a threshold of about 1/3 of the total image, the final mask is simply the difference mask. This method produces a shaky real-time output, and may be the reason for the poor segmentation in the final result. Research into methods for other reconstruction techniques is required for a better result.

Finally, a metric, other than visual inspection is needed to judge the merit of each technique for this application. The Jaccard index was suggested but a sufficient implementation was not completed in time.

VII. Conclusion

A MATLAB toolbox of programs is developed to provide a solution for the application of Chroma keying in single user video conferencing. With the use of background subtraction techniques, and motion segmentation via optical flow techniques, a user is able to change the background of the video conferencing stream. The toolbox is nascent, and produces poor real-time results. An improved combination of optical flow results, and more accurate background estimation and subtraction is needed to make this toolbox useful as an everyday tool.

Acknowledgment

I would like to thank my mentor Mina Makar and the course TA, David Chen for all the help and guidance they both offered the course of this project. Given the time frame of the course project, their aid was paramount in making this project tractable in the time allotted.

REFERENCES

[1] M. Piccardi, “Background Subtraction Techniques: a review,” in Proc. of IEEE SMC 2004 International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3099–3104, Oct. 2004.

[2] C. Stauffer and W.E.L. Grimson, “Adaptive background mixture model for real-time tracking”, CVPR99, Fort Collins, CO, June 1999.

[3] N.J.B. McFarlane and C.P. Schofield, “Segmentation and tracking of piglets in images”, in Machine Vision and Applications, vol. 8, pp. 187-193, May 1995.

[4] P. Dollár, Piotr’s Image and Video Matlab Toolbox, http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html

[5] B.D. Lucas and T. Kanade, “An Iterative image registration technique with an application to stereo vision”, in International Joint Conference on Artificial Intelligence, pp. 674-697, 1981.

[6] J. Shi and C. Tomasi, “Good features to track”, in IEEE Conference on Computer Vision and Pattern Recognition, pp. 593-600, 1994.

[7] B.K.P. Horn and B.G. Schunck, “Determining optical flow”, Artificail Intelligence, vol. 17, pp. 185-203, 1981.

[8] S. Benton, “Background subtraction, part 1: MATLAB models”, unpublished, http://www.dspdesignline.com, Dec. 2009.