Real Time Head Tracker using Frame grabber and a Webcam

Venkata Pradeep Paruchuri, Subhash Pedamallu
Department of Electrical and Computer Engineering
Clemson University
Clemson, South Carolina 29631
pparuch, speddam @clemson.edu

Abstract

We present a simple prototype system for real time tracking of a human head. This system uses a simple yet a effective tracking algorithm presented in [3]. The general requirements of a real time tracking algorithm – it should be computationally inexpensive, should possess the ability to perform in different environments and should be able to start and initialize itself with minimum knowledge about the environment, [6] are well addressed by the elliptical head tracking algorithm. Our tracker has been developed on two different configurations - USB Webcam and Frame Grabber both running on Linux. Frame grabber is a device, which can seize and record a single frame of video information out of a sequence of multiple frames [1]. The Bt848 chip is a very common solution for Frame grabbers. There are two types of device drivers in Linux available for the bt848 based frame grabbers bttv – where the images are directly sent to video memory and bt848 – which is more suitable for image processing applications as it allows Direct Memory access of images into main memory [2].The system tracks the objects and displays the resulting tracked imagesreal time on the computer screen. The object is modeled as an ellipse. Frame rates close to about 5 frames per second have been achieved.

Introduction

Real time tracking has been gaining a lot of interest among computer vision and image processing researchers because of its potentially wide range of applications which not only include robotics – incorporating vision in robots and surveillance but also various other fields like sports, entertainment industry etc. Since it is not feasible to interrupt a tracker to correct tracking errors by human intervention, an algorithm used for real time tracking should be robust and reliable. Also the algorithm must be computationally inexpensive to achieve good frame rate – which is very important in some areas like tracking athletes in live sport events where the loss of frames would result in the loss of vital information for later analysis by sports experts [6].

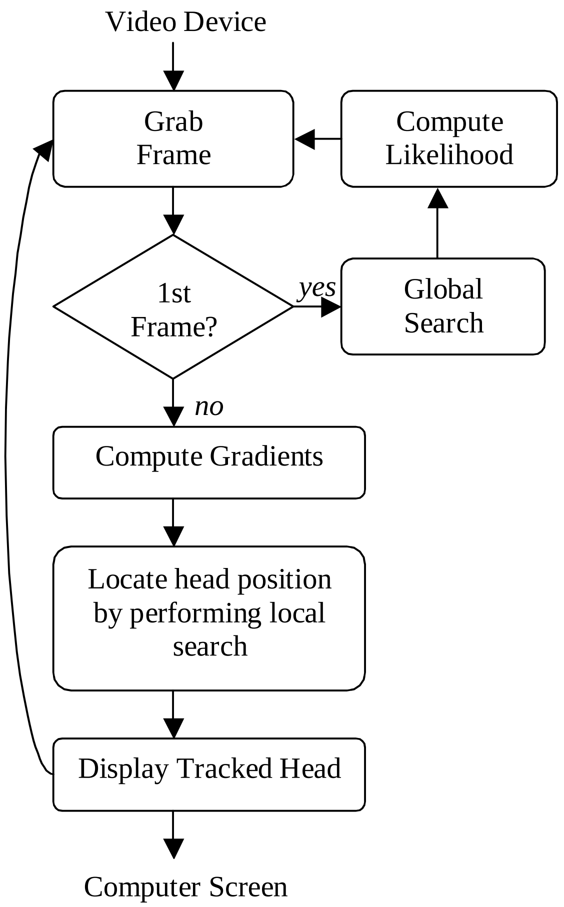

Fig 1. Block diagram of the Real time tracker

The overall system is as depicted in the block diagram given in figure 1. The image acquisition module grabs frames from the video device which are fed into the tracking module. Now the tracking module checks if the grabbed frame is the first frame. A global search is performed for the first frame and the likelihood is computed. This way the tracking module automatically initializes the tracker by performing this global search on the first frame thus eliminating the need for any explicit knowledge of the environment before hand. For all the subsequent frames, the location of head is found by performing a local search in the search range. Finally the object which is tracked in real time is displayed on the Computer screen.

The material in this paper is organized as follows – first we discuss the concept of acquiring images from the video devices – Webcam and a Frame grabber. Next we discuss about the details of the Tracking algorithm used – Elliptical Head Tracker [3]. We then present the real time implementation of this tracker and discuss the implementation issues. Finally the experimental results are reported and conclusions followed.

Image Acquisition

This section discusses about acquiring images from two different video devices – Webcam and a Frame grabber.

Frame Grabber - Frame grabber is a device, which can seize and record a single frame of video information out of a sequence of multiple frames. The image data is stored in the memory buffers of the Frame grabber, from where we can read the information by Direct Memory access.

Our system uses the Bt848 based frame grabber which is connected to a video source. Bt848 is a low cost single-chip solution by Brooktree, now Connexant Systems for analog NTSC/PAL/SECAM video capture on the PCI bus. There are several Frame Grabbers which use the bt848 chip and all run on the same device driver [2]. While the Bt848 driver was originally developed for the FreeBSD operating system this was ported under Linux by Brad Parker. The user interface is the same as in a meteor driver so that applications can be written with either a Bt848 based frame grabber or a Matrox Meteor Card. The meteor driver allows for DMA transfer of images into main memory and is very efficient for image processing [7].

It takes advantage of the PCI based system's high bandwidth and inherent multimedia capability. It is designed to be interoperable with any other PCI multimedia device at the component or board level, thus enabling video capture and overlay capability to be added to PCI systems in a modular fashion at low cost. Further the Bt848 solution is independent of the PCI system bus topology and may be used directly on a motherboard [8].

Webcam – A webcam consists of a digital camera which is connected to a piece of hardware used to grab images from the digital camera at regular intervals.

The webcam used is a Logitech Quickcam Notebook Pro which connects to the USB port. Linux 2.6 Kernel provides a device driver for this webcam which can be used to interact with the device.

We developed separate applications in C for grabbing images from the Bt848 frame grabber, and the webcam based on their Linux device drivers.

Elliptical Head Tracking Algorithm

The head contour is approximated by a two dimensional model –Ellipse with an aspect ratio of 1.2. The basic idea of the algorithm is to perform a local search of the image where the sum of the normalized image gradients computed over the boundary pixels of an ellipse is the maximum. Velocity prediction is further added to eliminate the limitations due to the maximum velocity conditions. It has been proved to be efficient for complete 360 degree rotations and supports reacquisition of the image and performs very well to handle occlusion, tilting, rotation, scaling problems and textured background. Prior work done did not support out of plane rotations, or employed facial color recognition, a background subtraction which is not suitable owing to the variations in the camera position and lighting conditions [3].

s is the ellipse state defined as (x, y, l), where x, y define position of the ellipse and l defines the scale of the ellipse. The optimal position (s*) is given by

$$s^* = \arg\max_{s \in S} \left\{ \frac{1}{N_l} \sum_{i=1}^{N_l} | g_i | \right\}$$

where N_l is the number of points on the perimeter of the ellipse, g_i is the gradient at pixel 'i'. The search space S is the set of all the states within some range of the predicted location.

$$S = \{ s : | x - x^p | \leq x_r , | y - y^p | \leq y_r , | l - l^p | \leq l_r$$

Where we used $x_r = y_r = 4$ and $l_r = 1$ in our implementation
The predicted state (x^p, y^p), the predicted scale l^p are obtained from the velocity prediction formulae as given below

$$x^p = x_{t-1} + (x_{t-2} - x_{t-3})$$
$$y^p = y_{t-1} + (y_{t-2} - y_{t-3})$$
$$l^p = l_{t-1}$$

The tracker's sensitivity to dominant backgrounds can further be eliminated if instead of just finding the gradient intensities, the square of the dot product of the gradient and the ellipse normal is computed [10]. It is different from other algorithms in that – gradient is summed over the entire parameter instead of just a few select points, and also it allows all of the date to be examined before a decision is made [3].

Prototype Tracker Implementation

The prototype tracker was implemented on two machines – one using a M-system's disk-on-chip running Linux 2.4 and the other one is Pentium Centrino based laptop running on Linux 2.6. We tested the Frame grabber version on the first machine and the Webcam version on the other one.

The Image Acquisition module and the Tracker modules are completely implemented in GNU C. All the routines used in the Tracker module have been completely developed from scratch without using any open source libraries. We used Matrox Meteor device driver provided by Prof. Adam Hoover to interact with the Frame Grabber and Phillips webcam driver provided with the Linux 2.6 kernel, to interact with the webcam.

The Image Acquisition module for the webcam involves conversions from YUV color space to RGB color space, since the Logitech webcam gives the image in the YUV format.

In the Tracker module - the gradients for the Red, green and Blue pixels are computed separately and the average is mapped on to form a gradient map of the image. An optimal position of the head is found by performing a global search on the first frame. The optimal position is where the sum of gradients over all the points of the ellipse boundary is the maximum. For all the subsequent frames the search area is restricted to a finite range forming the likelihood where the object may be present. Now the tracked frame along with the ellipse mapped on to the head is displayed.

For frame grabber setup, the monitor is configured to 16-bit display, but frame grabber gives us the RGB24 image so to display images we are employing Pixel packing. Pixel packing technique takes original RGB24 image, and selects most significant bits of each – 5 from Red, 6 from Green, and 5 from Blue components to give us 16-bit image data.

Frame rates up to 4-5 Frames per second have been achieved on the Webcam setup and 2 frames per second on Frame grabber setup. Further optimization is possible by parallelization of the computations. The use of color histograms is expected to further increase the efficiency of tracking by making it insensitive to textured backgrounds [5]. Some algorithms also suggest employing methods like background subtraction, thresholding etc. for enhancing the performance [6].

Experimental Results

The tracker was tested using the setup described above under different environments and lighting conditions and it has been observed that it performs better in plain background environments even with occlusions, rotations and

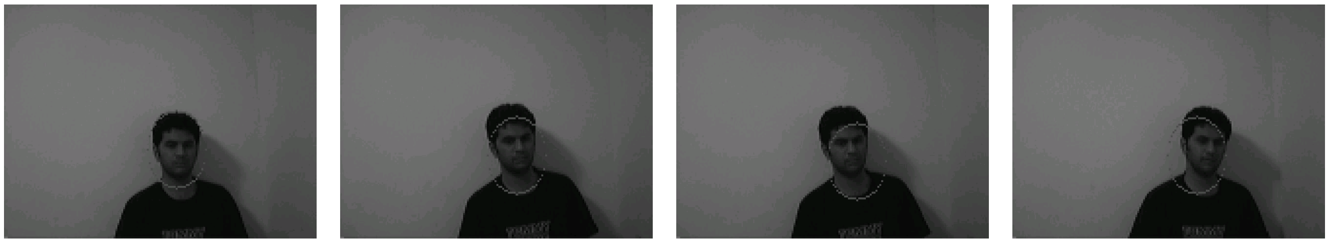
Fig 2 Tracked Images – Untextured Background

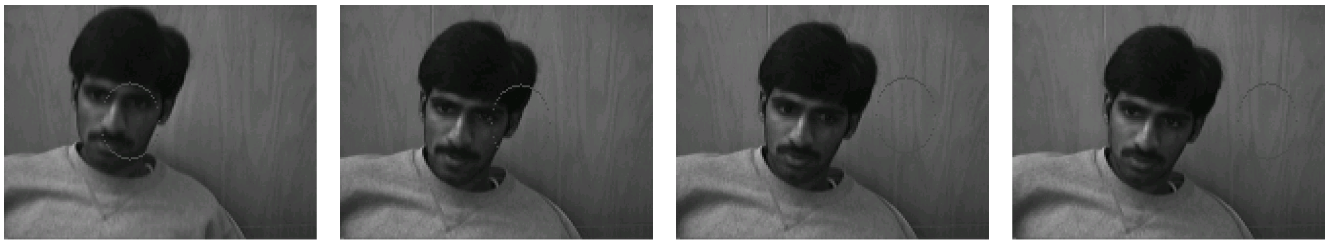
Fig 3 Textured Background –Gradient Intensity

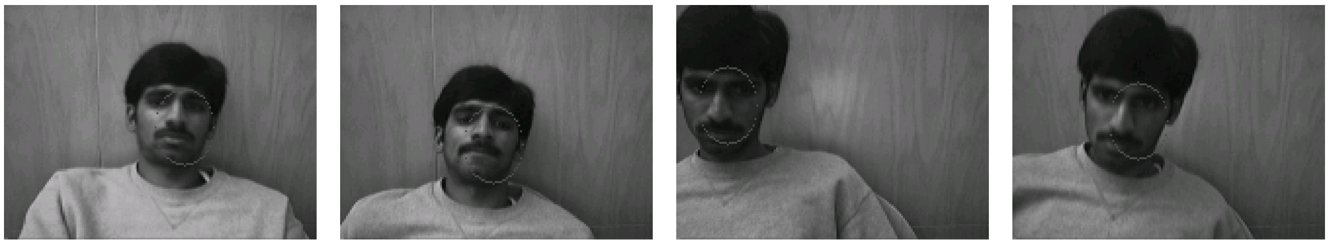
Fig 4Textured Background –Normal Dot Product

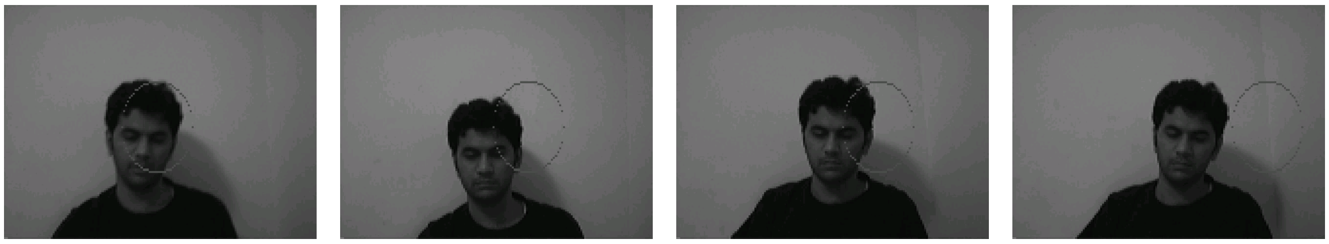
Fig 5 Non Uniform Lighting Conditions

scaling. This tracker occasionally looses track of the subject when performing in environments with non uniform lighting conditions and textured backgrounds. *Fig 2* shows some frames taken from the tracking process under uniform lighting conditions and untextured background. Figure 3 shows the tracked images where the tracker fails due to textured background. Figure 4 shows that adopting gradient normal dot product [10] solves this. Figure 5 shows the tracked images, where the tracker fails due to non uniform lighting.

Conclusions and Future Work

Real time tracking using a Frame grabber and a webcam has been tested with the Elliptical head tracking algorithm. Frame rates close to about 5 frames per second have been achieved. The tracking accuracy is satisfactory even when the head is rotated out of plane thus eliminating the limitations of the tracking algorithms which depend on the facial color. It works fine even with textured backgrounds although occasionally the algorithm fails when the gradient intensities of the background dominate that of the foreground head to be tracked.

The tracking algorithm can be further optimized to give a much higher frame rate. Currently frame rates up to about 30 Frames per second have been reported [9]. There is a loss of information in the form of frames on the temporal space when there is a low frame rate, which may not be desirable. Frames rates upto 10 Frames per second can be achieved by optimizing the software, which is sufficient for real time tracking.

Our implementation does not perform very well in the case of environments with non uniform lighting conditions, or dominant textured backgrounds. This can be further improved by employing algorithms like feature tracking etc.

Acknowledgements

Thanks to Dr. Stanley Birchfield for giving us this opportunity to work, and his constant help during the course of this project. Special thanks to Dr. Adam Hoover for letting us use the equipment in his lab. Also thanks to our friend Kamal Lanka for providing us the Logitech webcam.

References

[1] Bandwidth Market, Glossary http://www.bandwidthmarket.com/resources/glossary/F5.html

[2] Bt848 Linux Device Driver, http://www.dis.uniroma1.it/~iocchi/bt848/

[3] Stanley Birchfield, “Elliptical Head Tracker”, *31st Asimolar Conference on Signals,Systems and, Computers, November 1997.*

[4] PPM Format, http://netpbm.sourceforge.net/doc/ppm.html

[5] Stanley Birchfield, “Elliptical Head Tracker using Intensity Gradients and Color Histograms”, *IEEE Conference on Computer Vision and Pattern Recognition, June 1998.*

[6] Janez Pers et. Al., “ A Low cost Real Time Tracker of Live Sport Events”

[7] Matrox Meteor Capture Card Driver Announcement,http://lists.freebsd.org/pipermail/freebsd-announce/1995-August/000086.html

[8] Digital Infotainment, Bt848 Single chip Video Capture card and PCI Bus Master, http://www.clavis.ne.jp/~listcam/bt848.html

[9] Carlos Morimoto, “ Real time detection of Eye and Faces”, *IBM Almaden Research Center*

[10] Nishihara, Personal Communication, Reported in Stanley Birchfield, “Elliptical Head Tracker”, *31st Asimolar Conference on Signals, Systems and, Computers, November 1997.*