Step Counting on Android Platform

Wei Chen
wechen@stanford.edu

***Abstract*— this report describes the algorithms and implementation of step counting using image processing techniques on Android Platform. The user can use an Android device to video take the subject walking between two points. The program will upload the video to a Matlab server. The algorithm implemented in Matlab will analyze the frames by extracting the human subject in the video frames and measure the area change pattern of the human shape. The number of local minimums is computed and is considered to be the number of steps. More image processing techniques will be discussed for more complex environments.**

Keywords: image processing; Android; segmentation;

I. INTRODUCTION

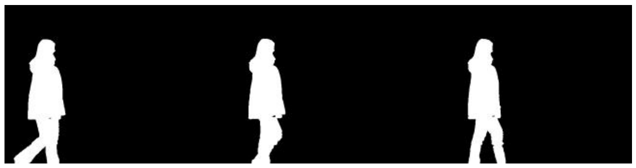
Smart phones like Android and IPhone are becoming more and more popular nowadays. These phones have the programmability like our regular PC and are also normally equipped with cameras, GPS and connected to the Internet through WIFI or 3G etc. More importantly, they are extremely portable and can be carried around wherever the user goes. This opens the doors for the application programmers to write many utilities that won't be possible before. The Google play and Apple App stores have tons of such apps and give the smart phone users conveniences and boost of productivities. Among all of these apps, there is a huge interest in utilizing the onboard camera for various tasks. The early ones like bar code reader, shopping assistant and mobile fax etc. have made the phone a versatile device more than a phone. In this project, the goal is to write an app that can count the number of steps of human walking and thus can be used for things like distance measurements. The motivation is that the current camera can easily have 30 frames per second frame rate and can capture every step of human walking or even fast running. The video taken then can be analyzed by the image processing algorithms to come up with the number of steps. This task however is not easy for human brain and eyes. The difficulties for this app are at the image processing steps depending on the how complex the surrounding environment is.

II. PREVIOUS WORK

The related previous works that are related to this project includes image segmentation, human detection. For the image segmentation, we use the Otsu's method and locally adaptive thresholding. For human detection, we study the possibility of using histogram of gradients method.

III. APPROACH

The user will use the smart phone to record a video of human walking from point A to point B. For the first version of this app, it is required that the video is taken from the side of the subject, that is, the video should clearly show the leg movements change. This is critical for the proper working of this app become the way we compute the steps is by tracking the human shape area change patterns. The theory is that when people walk, they first land one foot on the ground and move the other leg from behind forward. So at one point during this period, two legs will overlap each other when observing from the side. Thus the human shape areas in the video frames will show a periodic pattern. This is showed in the three frames below. The frame in the middle has less white area than the other two as the two legs overlap most.



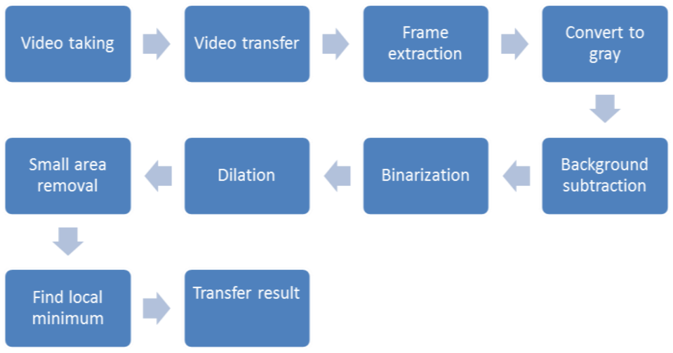
The user needs to make sure the first few frames of the video consist of only the background. This is to facilitate the background subtraction. The algorithm will use the first frame as a background and this also requires that the user makes sure the smart phone is in the static state when doing the video recording. Otherwise the background subtraction will be off. This second requirement can be replaced by more sophisticated algorithm such as image alignment. For this simple prototype system, we will use the simple point subtraction method so this is required.

After the video is taken, the first step is to transfer the video from the mobile device to the Matlab server. This cloud computing scheme is quite popular in the real work application too. Even though mobile devices nowadays have quite impressive computing power but running complex algorithms such as image processing algorithms will still cause too much delay for real time interaction and power consumption for long battery life.

After the video is push to the server, the Matlab script will read in the video. The first step is to sample a smaller number of frames for analysis. This is needed because the video is taken at 30 frames per second and this is too much for analyzing the walking human subject. Based on the moving speed of the subject, the sample rate can be changed accordingly. Then the video frames are changed to gray scale frames as it is easier to apply image processing algorithms on gray scale images. The next step is to obtain the background frame. As mentioned at the beginning of this paragraph, the first few frames are taken with the scene empty so can be used as the background reference. Then the background frame is subtracted from all the sampled video frames. The resulting video frames are gray scale images with human shapes in them. Then we use Otsu's method to change the gray scale images to binary images with foreground and background. The human shape will be represented by white pixels while the rest of the area is consisted of black pixels. Then dilation is applied to the binary images. This is because when we subtract the background from the video frame, the human shape will have small holes or thin lines in them. Dilation can fill the holes or the lines. After this stage, the processed images are almost ready for the next step. According to the video recording environment, other filters can be used to make the image cleaner. For example, if there are small unrelated areas scatter in the image, a filter can be used to clean up the small areas.

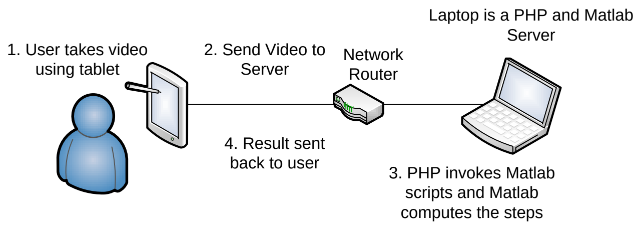
The last step is to compute the number of steps from the processed images. The method we use is to calculate the area of the human shape and find the local minimums of the area. The logic behind it is explained in the first paragraph in this section.

Then result is sent back to the mobile device for display. The overall approach is depicted by the diagram below.



IV. IMPLEMENATION

The prototype system was developed on an Android device, Asus Transformer Prime and a Windows 7 laptop PC. The system has client software running on the tablet that manages the video recording and result display and server software that trigger the image processing and result reporting. The communication between them is thru wireless network. The overall system architecture is depicted in the diagram below.


1. User takes video using tablet
2. Send Video to Server
Network Router
Laptop is a PHP and Matlab Server
4. Result sent back to user
3. PHP invokes Matlab scripts and Matlab computes the steps

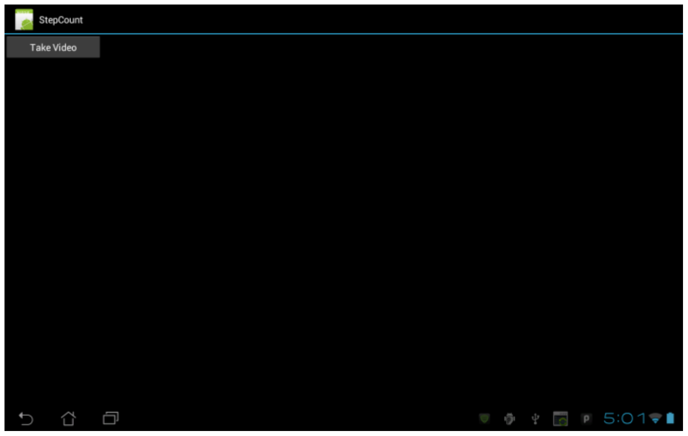
We explain the implementation for each component below.

1) *Client Software*

We implement the client software by referring to the camera sample code on the Android developers' website. We use the Intent method to implement the video recording functionality. So the app will use the existing camera app on the device to take the video. In this case, the existing camera app we use is the default Android Camera app. This allows us to set the resolution of the video, flash light mode, white balance etc. This is useful as it allows us to pick the best configuration for our purposes. In our testing, we use the lowest resolution video which is 480p video. This is good enough for our analysis and at the same time save the video transfer bandwidth.

The workflow of the client software starts when the video record button is clicked. The software then starts the video recording. After the user clicks the stop button, the video is saved to the device and the path is passed to the server task thread to start the transfer process. The server task thread will setup the http connection with the server and transfer the file. After the video file is transferred successfully, it waits for the output from the server. Then it displays the number as a Toast message.

The screen shot of the UI is shown below.



2) *Server software*

The server acts as PHP server and Matlab server. As a PHP server, it runs the countStep.php script that receives the video file, sends back the result to client and triggers the other commands on the server. The php script triggers two commands in our case. One is to call the video format converter utility to convert the mp4 format to avi format. This is required because the Matlab version on the server does not accept mp4 file. The other command is to run the matlab script.

3) Matlab script

The core algorithm is implemented in the countStep.m Matlab script. It implements the algorithms we described in the last section. There are a few steps that worth mentioned.

The first one is about finding the background frame and skipping the background frames. We currently require the user to take the first second of video as background frame. Thus about the first 30 frames will be background frames. Obtaining the accurate background image will make the following segmented images look nicer. But it is not that critical for the step counting as all the video frames will refer to the same background so a slightly off background will not cause too different results.

The second one is about finding the starting frame for step counting. Our method requires the user first takes the background scene and the subject moves in the camera view. There are frames that human subject is partially in the picture. In order to skip those frames, our algorithm searches the frames and finds the first local maximum then uses that as the starting frame.

The last part of the code is to save the result in a text file and store it in a location that php script will check and send back to the mobile device.

V. FUTURE WORK

There are a couple of improvements that can be done to the algorithms and implementations for a more robust and user friendly system.

The current implementation requires the mobile device to be static during the video recording. This means the mobile device need a rack or other kind of holder to stabilize it. This is not quite practical in most of the cases as the users prefer to be able to shoot the video with hands. The shaking of the hand can cause the background change constantly. So the simple point subtraction method will not work. An image alignment algorithm may be applied to alleviate the problem here.

Another possible improvement we can make is to incorporate the human detection algorithm. Our current implementation relies on background subtraction to work properly and assumes that remaining subject is human. This may not always true depending on the stability of the video taken and the background scenes. By using human detection, the algorithm will know which area is really human shape and can compute the proper areas. The related algorithm is histogram of gradients (HOG) etc.

The current implementation first converts the video to gray scale frames. This can cause the segmentation problem if the background happens to have very similar gray value compared to human subject. A more robust solution will be to use all the three color channels for segmentation.

ACKNOWLEDGMENT

We would like to thank Prof. Girod for his excellent teaching for this difficult subject. We also want to thank CAs David and Derek for their clear explanations on homework problems and quick responses on the course forum. Last but not least, we would like to thank Mina for his help over the process of this project.

REFERENCES

[1] M.Bichsel, “Segmenting Simply Connected Moving Objects in a Static Scene,” Trans. Pattern Analysis and Machine Intelligence, vol. 16, no. 11, pp. 1,138—1,142, Nov. 1994.

[2] K.Rohr, “Towards Model-Based Recognition of Human Movements in Image Sequences,” CVGIP:Image Understanding, vol.59, no.1,pp.94—115, Jan.1994.

[3] Chin-Chun Chang; Wen-Hsiang Tsai; , "Vision-based tracking and interpretation of human leg movement for virtual reality applications," Circuits and Systems for Video Technology, IEEE Transactions on , vol.11, no.1, pp.9-24, Jan 2001.

[4] Dalal, N. Triggs, B., “Histograms of Oriented Gradients for Human Detection,” IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 005, VOL 1, pages 886-893

[5] http://developer.android.com/index.html

[6] http://www.stanford.edu/class/ee368/Android/index.html.