Accelerating Real-time Face Detection on a Raspberry Pi Telepresence Robot

Krit Janard and Worawan Marurngsith
Department of Computer Science
Thammasat University
Pathum Thani, Thailand

Abstract— **Effective face detection in real-time is an essential procedure for achieving autonomous motion in telepresence robots. Since the procedure demand high computation power, using it to create autonomous motion in low-cost robots is a challenge. This paper addresses this issue and making three contributions. First, the process to enabling the real-time face detection on Raspberry Pi's graphical processor is presented. Second, the development of an autonomous pan-tilt telepresence robot to follow an interlocutor face using two Raspberry Pi-1 model B is demonstrated. Third, the evaluation on resource requirements when operating the robot in various scenarios is described. The face detection module ran in average at 16.7 Quarter VGA frames per second, while mediating real-time video conversation remotely between two parties. The results confirmed that vision-based autonomous motion can be added to a low-cost telepresence robots with acceptable performance. Thus, making secure telecommunication via robots is viable with less budget constraint.**

Index Terms—**Face detection, LBP, robot, Raspberry Pi, GPU.**

I. Introduction

Telepresence robot is an invention for making users feel as if they were in two places at the same time. The user controlled his/her avatar robot remotely, while being in another place. Telepresence robots equipped with video conferencing module. Thus, the user can interact with his/her interlocutor at the robot site. Various types of telepresence robots are available for commercial, in both mobile and desktop features. Apart from using as a tool for scientific discovery in unmanned area, telepresence robot usage has gained advantage over video conferencing in wide range of applications *e.g*. for research, elderly health care, office, school and general purpose uses [1]. The usage of telepresence robots are still limited as commercial robots are not affordable. Emerging of the ultra-low cost computer for education [2], the Raspberry Pi (RP), has brought attention on telepresence robot research. As the RP computer is a platform aimed for teaching computer component in the DIY fashion, it has a limited computation power. A research on telepresence robot has used the RP computer for controlling the robot motion and offloaded some computation on the Cloud [3]. Using this technique, the robot has been equipped with some autonomous features like following an interlocutor face.

Real-time face detection is a key step in controlling the pan and tilt unit (PTU) of telepresence robots to automatically follow the face of an interlocutor. By doing so, the robot autonomy is enhanced; and it could be extended to maintain eye contact with the interlocutor. Several research have shown that this feature increases users' satisfaction when using telepresence robots in office environment [4-6]. Some research has proposed techniques to add more user interaction capabilities to the robot using the Kinect camera, e.g. implementing motion tracking and image-based face tracking [5]; having gesture based recognition [6]. The key success factor for face tracking and face recognition tasks in the robots is the effective face detection technique used in real-time.

Several real-time face detection techniques have been used in high performance telepresence robots [1, 7], but has not been mentioned in a robot using pure RP computers. Tripathy and Daschoudhury [8] presented a technique for face tracking using the Haar classifier on the graphical processors (GPU) of the RP computer. Bilaniuk et al. [9] has shown that the face detection can run at 5 VGA frames per second when highly parallelized the Local Binary Pattern (LBP) technique on an embedded SIMD architecture with low power consumption. Experiments from various algorithms on the RP computer confirmed the effective of LBP [10]. A research has shown that different applications can be accelerated by offloading to the GPU of a RP computer [11]. Existing research have confirmed the feasibility to implement effective real-time face detection on the RP computer. However, to use the face detection for moving the PTU of a pure RP telepresence robot to follow a face is still a challenge.

This paper presents a technique to enable the real time face detection on the RP's GPU and using the result to control the PTU of a pure RP telepresence robot. The robot is connected remotely with its user using a client application on Android smartphone. The user can set the robot to follow a face of the interlocutor who stay at the robot site; or manually control the PTU of the robot by turning or rotating the smartphone. The robot mediates a real-time video conference between the two parties. For this, the proposed robot uses two RP computers model 1B as the main processors. The paper presents the key concepts used for implementing the robot and makes three contributions.

1) A pipelined technique used for accelerating the real-time face detection on the RP's GPU is presented (Section II).

978-1-4673-7551-1/15/$31.00© 2015 IEEE

The technique uses Local Binary Pattern (LBP) algorithm available in the OpenCV library, which achieved 16.7 Quarter VGA (QVGA) frames per second (Section IV).

2) The integration of the face-detection technique to allow the head of a telepresence robot to follow an interlocutor face is demonstrated (Section III).
3) The evaluation on resource requirements when operating the robot in various scenarios is described (Section IV).

The experimental results show that the proposed face detection and following techniques used in the robot requires only 10 percent more memory, approximately 10 MB, in comparing to operating the robot in the normal mode. The results also highlighted that using the ServoBlaster library to control the PTU of the robot increased more than 10 percent of CPU resource and only achieve 15 QVGA frames per second speed. Thus, different hardware should be used. For all testing scenarios, the robot can follow a face of the interlocutor at real time (16.7 QVGA frames per second in average), while mediating real-time video conversation remotely. Thus, vision-based autonomous motion can be added to a pure RP telepresence robot with acceptable performance.

II. Real-Time Face Detection on RP-1B computer

Real-time face detection has been implemented on a telepresence robot which has two RP computers, model 1B. As shown in Fig.1, the face detection module is allocated on the RP#1 computer, while the real-time video conferencing is handled by the RP#2 computer. The robot has two camera, *i.e.*, the RP camera module for face detection and the web cam for video conferencing. Existing research has shown the benefits of using LBP algorithms and offloading the face detection to the RP's GPU. Thus, the LBP algorithm from the Open Source Computer Vision (OpenCV) library has been used for face detection. To offload the computation of the face detection process to the CPU, the multi-media abstraction layer (MMAL) application program interface is used.

A. Face Detection Technique

Baseline face detection techniques used in telepresence robots are the Viola-Jones detector, Haar classifier and LBP [12]. In this work, the LBP algorithm is used as research confirm its advantage of speed over the other two techniques. Despite, the correctness of results depending on the effects of illumination. The LBP is used for measuring local image contrast. The whole image is broken into small parts called integral image, which is normally 3x3 pixels. First, a binary threshold window of 3x3 pixel is calculated for each integral image. The value of the center pixel is compare with all eight neighbors. If the value in the neighbors' pixel is greater than or equal to the center pixel, the threshold bit at is set to 1; otherwise is set to 0. Second, the 3x3 threshold windows is multiplied with the 3x3 weighted window, resulting the weighted threshold 3x3 windows. The weighted window is normally two power by the position of the neighbors. Third the LBP value, the value of the central pixel, is calculated by the summation of the neighbors' weighted threshold value.

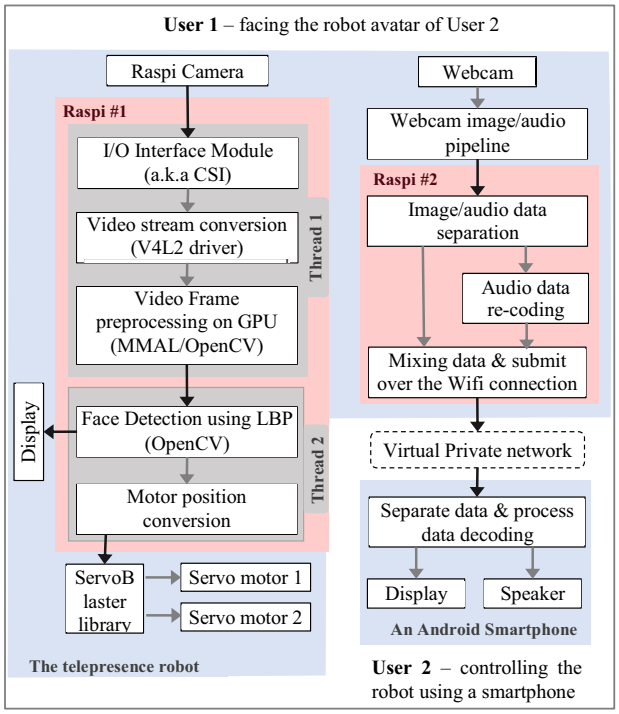

Fig. 1 Process of the real-time face detection model and the real-time video communication with the connected smartphones.

B. Real-time Face Detection Pipelining on Raspberry Pi

Implementation of the LBP algorithm is available in OpenCV. To accelerate the face detection process in the GPU of the RP computer, the pipelined implementation has been used. As shown in Fig. 1 (left), the face detection was implemented in two asynchronous threads, using the MMAL API. The first thread is responsible for obtaining and preprocessing image frame obtained from the RP camera module. Once an image frame is ready the callback function to perform face detection is signaled. The second thread implements this callback function, which is responsible for calling the LBP algorithm and calculating the motor position. The second thread interfaces with the display to show the face detected (if mirroring mode is set). It also interfaces with the ServoBlaster library to pass the x and y position to the corresponding servo motors attached in the PTU.

OpenCV has a set of pretrained object-recognition files which can be used for detecting objects which has consistent texture. A set of frontal face models from these pretrained objects are used to train the face detection module by specifying the corresponding *xml* file. Thus, prior to running the robot, thirty pretrained faces were used to calibrate the face detection process.

It is important to note that the RP camera module used in the robot has been set to support only the gray scale image. The default QVGA video frame size (320x240) is chosen. Two level of frame speed *i.e.*, 15 frames per second (fps) and 30 fps are used to exercise the robot. It is possible to configure the video frame size to VGA (640x480). However, testing on the VGA frame size is not yet covered in this initial stage.

TABLE I. SPECIFICATION OF THE ROBOT

Physical Specifications	
Pan and Tilt	5Kg-cm Max Torque, Up to 0.21 second per 60 degree Max Speed
Maximum Degree	180 degree Pan, 180 degree (lock for safety at 120) Tilt
Computer and Display	
Processor	Raspberry Pi I model B Broadcom BCM2835 SoC 800MHz (OC) single-core
Memory	512MB, 8GB SD Card Class 10
GPU	Broadcom VideoCore IV
Screen	5 inch screen with 720x480 max resolution
Face Detection Camera	
Resolution	1440x1080 max, 640x480, 320x240 effective
Frame Rate	30 fps max, 25 fps effective
54 degree Angle of View, 1 m. to infinity Focus Length	
Streaming Camera	
Resolution	640x480 max, 640x480, 320x240 effective
Frame Rate	30 fps max and effective
Angle of View	58 degree
Focus Length	1m to infinity
Servo Controller	ATmega32u4 or ATmega328
Microphone Type	Mono
Electrical Specifications	
Power Input	100-250VAC 47-63Hz
Input leakage current	<0.7mA / 220VAC
Line regulation (full load)	+-0.5%
Output Power	50W
Output Voltage 1	12VDC 2A +-10%
Output Voltage 2	5VDC 4A +-0.5%
Efficiency	>85%

III. AUTONOMOUS PAN-TILT TELEPRESENCE ROBOT

The real-time face detection unit has been added to the development of a telepresence robot. This is to allow the robot to have autonomous PTU to follow the interlocutor face. The robot consists of a monitor, a web cam, an RP camera module, a ServoBlaster library and two servo motors, a power supply (as the prototype shown in Fig.2). As described in Section II, two RP computers model 1B have been used for processing the face detection and video conferencing. The specification of the robot is listed in Table 1. The total cost of the robot prototype is approximately 550 USD as of today currency value. For safety reason, the robot will be packaged using Acrylic cover which push the cost up to 600 USD.

A. Mode of operations

As shown in Fig. 2, the robot is built for facilitating the real-time video conferencing between two users. The first user (User1 in the figure) stays at the robot site and interacts with the robot. The second user (User 2, at the bottom of the Fig.2)

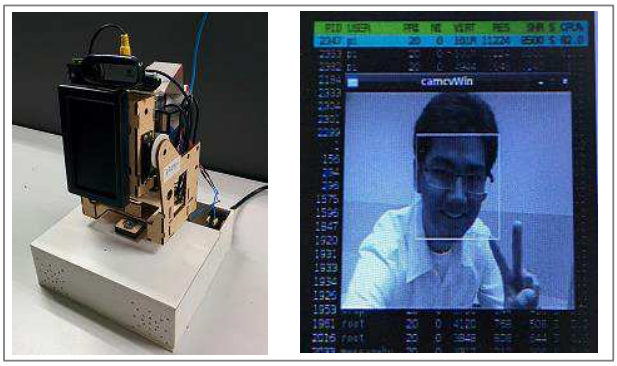

Fig. 2 The telepresence robot (left) prototype of the robot hardware design (right) the robot's screen when testing face detection module.

remotely connects to the robot and takes control of the robot by using an application provided for any Android smartphone. User 2 can configure the robot as his/her avatar using the following configurations.

Freeze: the robot has no movement.

Manual control: the robot is controlled remotely by User 2 moving his/her mobile phone. In this mode the Android application detect the movement of the mobile phone by suing the data from accelerometer and gyroscope sensors. The offset position is calculated and transferred back to the robot. Note that the RP#2 computer is also connected to the ServoBlaster library to control the PTU.

Autonomous Face following: this mode is used to allow the robot to move the PTU autonomously to follow a face. When using this mode, the robot will first detect a face at the center of the screen. Then the face detection is calculated in every second to detect whether the face has moved. This step is done by keeping the current frame with face position resulting from the LBP algorithm. At every second, the new frame is used to calculate LBP value; and the results are compared with the current frame and face position value. If a face has been detected, and moved, the displacement of the x and y axis are calculated. These result are then mapped back to the absolute position of the motor, and send to the ServoBlaster library. If the no face has been detected in the current frame but a face is detected later on, the motor position is used to calculate the offset with the new face frame. The new x, y values are calculated to keep the face frame in the middle of the QVGA frame. The updated x and y values are sent to the ServoBlaster library to move the PTU.

B. Video conferencing interface with Android smartphone

As shown in Fig.2 (right), the connection between the Android smartphone is done by the virtual private network (VPN) connection initiated by the application in Android, to the port listened by the second RP computer in the robot. Once the connection established, two sets of connections are done concurrently. First, the manual control of the robot PTU is done as a background process by using `AsyncTask`, an asynchronous lightweight thread. Second, the web cam is interface is done in foreground. The PTU position controlled is described earlier in the manual control mode.

As the web cam is connected via a USB port, the video and audio signal received has been processed by the webcam to the mixed video and audio data. This mixed data comprises the compressed video data using H.264 encoding and the compressed audio data using AAC encoding. Once the mixed data arrived at the RP computer, the data is preprocessed into the size and format that is suitable to display on the smart phone camera. First, the mixed data is separated back into the compressed video and compressed audio stream. The audio stream is decoded (using AAC decoding) and then encoded again using the Android specific audio compression. The recoding audio data is mixed with the compressed video data; and sent over the VPN. The Android application received the mixed compressed data and separate it back to video and audio stream. Both stream are decoded and send to the display and speaker. On the way back from the smart phone to the robot, the process is reverted. Thus, this video conferencing process is going on during until User 2 stops the communication and frees the robot.

IV. Evaluation and Discussion

The real-time face detection for adding face following feature in the pure RP robot has been evaluated by measuring the resource requirements. These include the memory usage, the CPU load, and the maximum send/receive bandwidth. The aim of the evaluation is to quantify the computation resources spent on face detection and observe the interference to the real time video conferencing module.

A. Experimental Method

Experiments have been done in four steps. First, the baseline resource requirement is measured while the robot is idle for ten minutes. The amount of memory usage and buffered, the percentage of CPU time, and send/receive bandwidth were collected. The collected data are summarized using the arithmetic mean (average). The baseline resource requirement is presented in Table II.

Second, the resource requirement during the robot is used for video conferencing and controlled manually is measured. The robot were used to communicate in using three content delivery modes *i.e.*, the audio only, the video only and the video and audio mode, without having PTU movement. As mention earlier, the audio quality used in the robot is 16-bit mono. The video quality can be adjusted using 15 fps or 30 fps, for each frame being a gray-scale QVGA size. When the audio and video content mode is used, the robot resource requirement at 15 fps frame rate is also measured when the manual PTU movement is activated *i.e.*, the ServoBlaster library was controlled by a remote user via smartphone. Raw data are collected in ten-minute time frame with three repeated run, sampling every 5 seconds; and are averaged. The results presented are the ratio between resource requirements during the video conferencing and the baseline (Fig. 3 – 5).

TABLE II. RESOURCE CONSUMPTION OF THE ROBOT IN TWO MODES

Mode of operations	Memory (MB) total 1 GB		CPU Time (percent)		Bandwidth Byte/sec (Bps)	
	Use	*Buff*	*User*	*Sys*	*Send*	*Receive*
Manual Control	74.5	31.8	12.4	35.3	2,769.2	1,649.6
Idle - baseline	72.5	31.5	4.9	7.5	695.2	244.5
Inc. Req. Ratio	2.8%	1.0%	2.5	4.7	4	6.7

Third, the resource requirement during the autonomous face following feature is turn on, while the video conferencing is ongoing. The robot were used to communicate in three scenario *i.e.*, when it is (1) idle, when it perform autonomous face following while users (2) using video with mute and (3) using both video and audio. Collected resource are summarized against the baseline resource requirements, in the same way they were done in the second step (Fig. 6 – 8).

Forth, the detailed performance of the face detection module is captured. The RP camera module was set to it maximum capability capturing video at 30 fps. The number of frame produced, the detection outcome, and the CPU loads is measured. Raw data are collected in ten-minute time frame with three repeated run, sampling every 5 seconds; and are averaged. The amount of frames was calculated per second, and the CPU load was calculated into percentage of overall CPU time. The results is shown in Table III.

Note that all measurement of memory and CPU time shown the resource used two RP computers. In this case, the memory usage was aggregated, and the parallel CPU time was measured using both RP computers.

B. Results

The baseline resource requirement, when the robot is idle, is shown in the highlighted row in Table II. In average, only 7 percent and 3 percent of internal memory was used and buffered[1]. The total CPU load in both user time (the time spent on executing the robot controlled program) and system time (the time spent on running operation system and services) was 12.4 percent. As the robot is connected with a remote user mobile phone, small amount of signal was exchanging to establish and keeping the connection. Because sending and receiving are not happen at the same time, in average the overall bandwidth required was dominated by the sending bandwidth of about 700 Bps. To quantify the increasing of resource requirements when operating the robot remotely, the ratio of increasing resource requirement with reference to baseline is calculated (the Inc. Req. Ratio row in Table II). The robot was controlled manually, and the resource requirement was collected. It was observed that in this situation, the increasing of memory requirement was not significant. The increasing of CPU loads and required bandwidth were 7.3 and 10.7 times with reference to the baseline. The manual control used nearly 48 percent of CPU time and required nearly 3KBps bandwidth.

[1] The 214 MB internal memory was cached in all experiments.

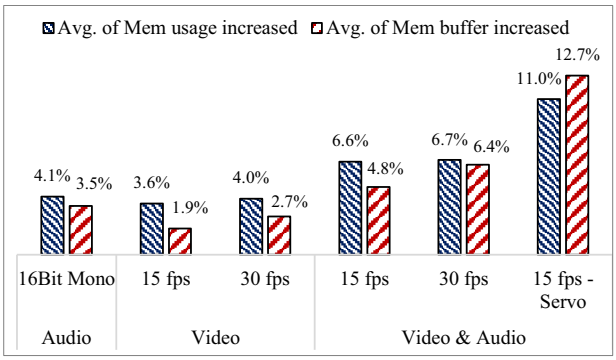

Fig. 3 Increased memory requirement for broadcasting different modes of content with reference to the baseline.

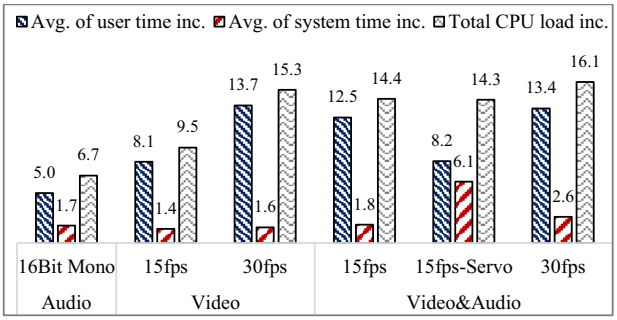

Fig. 4 Increased CPU loads for video conferencing only and with manual movement (Servo) (value shown in times higher than the baseline).

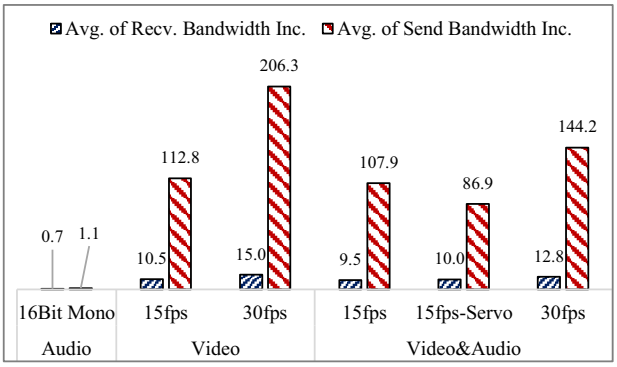

Fig. 5 Increased bandwidth requirement for video conferencing only and with manual movement (Servo) (times higher than the baseline).

The second step of the experiment is to quantify the increased resource requirement reference with the baseline while using manual controlled mode during video conferencing. The results of memory requirement (Fig.3) show that when no robot motion, 108 - 111 MB internal memory is required, ranging from 5.5 – 13.1 percent increase over the baseline (with 0.03 standard deviation). More memory is required for manually controlled the robot (Fig.3 15 fps-servo). The required CPU loads when no robot movement range from 38 – 86 percent of total CPU time, increase from 6.7 – 15.3 times over the baseline (Fig. 4). The user time dominates the CPU load; and is clearly increased when setting more fps at the web cam. More system time and CPU load required when manually controlled the robot with

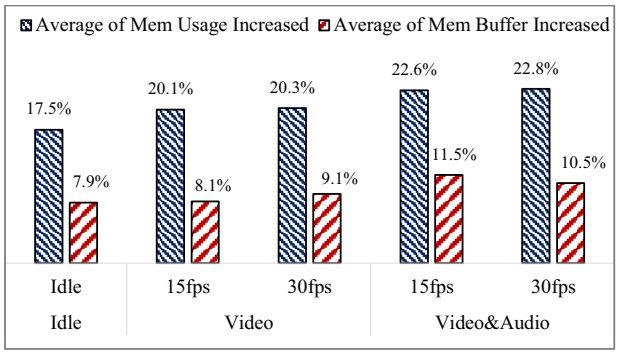

Fig. 6 Increased memory requirement when using face following during the video conference with different modes of content (reference with baseline).

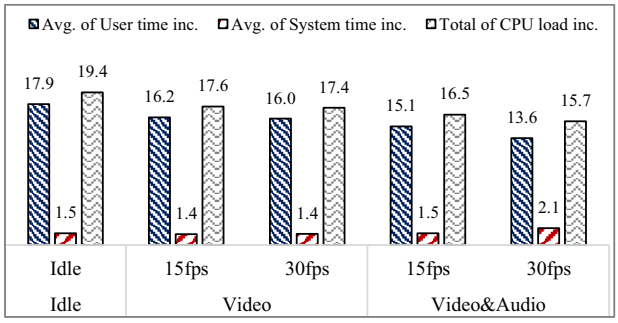

Fig. 7 Increased CPU loads when using face following during a video conference (value shown in times higher than the baseline).

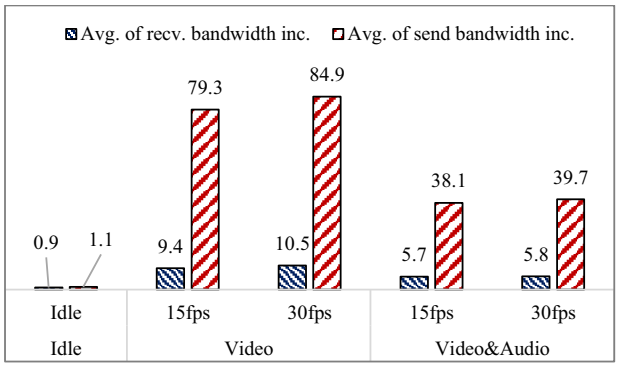

Fig. 8 Increased bandwidth requirement when using face following during a video conference (value shown in times higher than the baseline).

the ServoBlaster library. However, the overall CPU load is the same as when using full video conference at 15 fps with still robot. The bandwidth requirement was dominated by sending, and shown high variation. The send bandwidth range from 751 Bps – 143.4 KBps, increased from 112.8 – 206.3 times over the baseline (Fig. 5). However, the manually controlled robot show surprisingly low bandwidth used. This is because the RP#2 computer was busy with connection from the mobile phone. Thus, the less time spent on processing the video signals and sending data causes less bandwidth used.

The third step of the experiment quantified the resource requirement while the autonomous face following feature is turn on, during a video conference session. The memory

TABLE III. PERFORMANCE AND CPU USAGE OF FACE DETECTION

Face detection outcome	fps	User time ratio
Face Found		
Moving detected	16.7	86.7%
No moving detected	14.9	87.1%
No face found		
Moving detected	17.7	86.9%
No moving detected	17.5	86.7%

requirement range from 120 – 125 MB in all content mode, increased from 25 – 34 percent over the baseline (Fig. 6). The CPU load was push to peak when using face following feature, increased quite stably from 15.7 – 19.4 times over the baseline (Fig. 7). Similar to the second experimental step, the sending bandwidth was dominate, however, shown less variation. The send bandwidth range from 760 Bps – 56 KBps, increased from 43.8 – 95.5 times over the baseline (Fig. 8) during a video conference session.

The last experimental step quantified the performance of the face detection technique in terms of speed and CPU load. Table III shows that, on average, face detection ran 16.7 Quarter VGA (QVGA) frames per second and used 87 percent of CPU time. The worst case scenario for real-time face detection occurred when target face is found, but no moving detected.

V. DISCUSSION

The internal memory of two RP computers (1GB in total) is enough for all telepresence robot operations including the autonomous face following feature. Adding face detection module pushes the CPU load up to its maximum. Thus, when adding a new feature, like mobility, to the robot a separate processing resource is required. Using the ServoBlaster library for interfacing with servo motors added overhead on the CPU load, thus, causing slower response from the robot. In the future work, this hardware will be replaced with an Arduino controller. A clear pattern of bandwidth requirement has not yet found using this experimental design. The maximum bandwidth used in this experiment is 143.4 KBps, or 206 times increased over the baseline. Thus, all findings suggested that the proposed face detection technique has reach the maximum usage of computation resource on RP computer model 1B. To achieve a low-cost autonomous telepresence robot, the stable and high speed communication bandwidth and the separation on CPU loads using task parallelism are the key design factors.

VI. CONCLUSION

This paper presents a technique to enable real-time face detection on the RP's GPU for making autonomous face following in an affordable, pure RP telepresence robot. The design of the remote steering technique by using a dedicated RP computer to support the real-time video conference has been described. The proposed techniques have been evaluated by measuring the memory, processing time, and bandwidth requirements. The results showed that the proposed face detection implementation exploited full CPU loads with little extra memory required, and has no impact on the quality of the real-time video conferencing. This showed that vision-based autonomous motion can be added to a low-cost, pure RP telepresence robots with acceptable performance.

ACKNOWLEDGMENT

This project has been funded by the Faculty of Science and Technology, Thammasat University. We would like to thank Asst. Prof. Dr. Chaiporn Jaikaeo, Asst. Prof. Dr.Saowaluk Watanapa and Dr.Pakorn Leesutthipornchai for valuable comments and discussion. We thanks the discussion and lesson learned on the Raspberry Pi community website.

REFERENCES

[1] A. Kristoffersson, S. Coradeschi, and A. Loutfi, "A review of mobile robotic telepresence," *Adv. in Hum.-Comp. Int.,* vol. 2013, pp. 3-3, 2013.

[2] R. Heeks and A. Robinson, "Ultra-low-cost computing and developing countries: Raspberry Pi and its potential in the "global South"," *Communications of the ACM,* vol. 56, pp. 22-24, 2013.

[3] S. S. Prabha, A. J. P. Antony, M. J. Meena, and S. R. Pandian, "Smart cloud robot using raspberry Pi," in *the ICRTIT 2014*, 2014.

[4] L. Riano, C. Burbridge, and T. M. McGinnity, "A Study of Enhanced Robot Autonomy in Telepresence," presented at the The 22nd Irish Conference on Artificial Intelligence and Cognitive Systems (AICS), 2011.

[5] R. Berri, D. Wolf, and F. Osorio, "Telepresence Robot with Image-Based Face Tracking and 3D Perception with Human Gesture Interface Using Kinect Sensor," in *Robotics: SBR LARS Robocontrol,* , 2014, pp. 205-210.

[6] T. Keng Peng, Y. Rui, C. Yuanwei, H. Zhiyong, and S. Liemhetcharat, "Gesture-based attention direction for a telepresence robot: Design and experimental study," in *Intelligent Robots and Systems,* 2014, pp. 4090-4095.

[7] K. Tsui, A. Norton, D. Brooks, E. McCann, M. Medvedev, J. Allspaw*, et al.*, "Iterative design of a semi-autonomous social telepresence robot research platform: a chronology," *Intelligent Service Robotics,* vol. 7, pp. 103-119, 2014.

[8] R. Tripathy and R. N. Daschoudhury, "Real-time Face Detection and Tracking Using Haar Classifier on Soc," *International Journal of Electronics and Computer Science Engineering,* vol. 3, pp. 175-184, 2012.

[9] O. Bilaniuk, E. Fazl-Ersi, R. Laganiere, C. Xu, D. Laroche, and C. Moulder, "Fast LBP Face Detection on Low-Power SIMD Architectures," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, 2014, pp. 630-636.

[10] S. Fernandes and J. Bala, "Low Power Affordable and Efficient Face Detection in the Presence of Various Noises and Blurring Effects on a Single-Board Computer," in *Proceedings of the 49th Annual Convention of the Computer Society of India (CSI) Volume 1*. vol. 337, 2015, pp. 119-127.

[11] S. Sabarinath, R. Shyam, C. Aneesh, R. Gandhiraj, and K. P. Soman, "Accelerated FFT computation for GNU radio using GPU of raspberry Pi," in *Smart Innovation, Systems and Technologies* vol. 32, 2015, pp. 657-664.

[12] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition,* vol. 29, pp. 51-59, 1// 1996.