Human Behavior Tracking with Vision Heatmap Toward Smart Building System

Arko Djajadi, Putra Utama Jaya, Riza Muhida
Swiss German University, BSD City, Indonesia,
arko@sgu.ac.id

***Abstract*—Vision system has broad applications in diverse areas. One of such areas is the increasing interest of systematic monitoring of human movements in certain space, in restricted areas or even in open space such as on the shop floor or even on a certain path. Non-vision methods suffer from many inherent weaknesses, but vision methods demand more resources. The focus of this research is to produce a heatmap system that shows human traffic behavior using camera vision with real-time image processing on a low-cost miniPC or laptop in a typical smart building, where relevant parameters are closely monitored. A static background image is used as the base image and is compared with current video frames to give information about changes that happen in the specified area. These changes are processed further to obtain the path of human movement. Changes that happen in the path will change the RGB value of the path. The change of RGB values translates into color gradation. The color gradation result is added to the static reference background image to produce a real-time heatmap with changing cumulative color gradation. Busy areas will produce a brighter heatmap. The heatmap system is processed successfully both on a miniPC (Raspberry Pi 2 Model B, Cubie, Odroid) and on a laptop. The system also would add vision analytic values to a simple surveillance or CCTV system with low cost.**

***Index Terms*—Vision; Heatmap; Thresholding; Gradation.**

I. INTRODUCTION

In this research, a Vision Heatmap System is used as a system to map human traffic occurring over a specific time in a certain space. Any type of human movement will be recorded and tracked. Recorded data will further be processed and give a result as a live picture that consists of gradation of color. The gradation of color shows the human traffic occurring over a specific time. By definition, heatmap is a visual or graphical representation of data using colors to indicate the degree of activity, where usually darker colors represent low activity, and brighter colors indicate high activity [1]. Both gray scale color or RGB color can be used for heatmap application, depending on the requirement and preference.

To track human movement, many types of sensors can be used such as motion sensors. However, motion sensors can only be used at a specific place, have limited coverage and have no visual indication. The placement of the motion sensors is not flexible and requires certain time to install all the sensors to cover relatively wide areas. Nowadays, technology such as cameras and vision have become more advanced. Almost every building now is installed with surveillance cameras. By using surveillance camera to obtain data, it is not necessary to install more sensors, as a single camera or even multiple cameras can cover certain areas effectively.

The main objective here is to develop a heatmap system by using camera vision. The system will try to find, track, and record the behavior of human movement in a specific area and in a specific time frame. Therefore, it is a reasonable decision to utilize the advancement in camera and vision technology in this research to achieve the final result of heatmap overlaid on a typical visual display.

Advancement in camera and vision technology includes both software and hardware development tools. One of the capable vision development tools is OpenCV [2] that can be run on a processing device such as a low-cost ARM-based miniPC [3]. The additional required hardware is a USB camera vision [4] that can easily be interfaced into a miniPC.

OpenCV (Open Source Computer Vision) is an open source library of programming functions for real-time computer vision that is hosted for legal download from GitHub repository. It uses a BSD license and is still free for both academic and commercial use. Many common programming languages such as C++, C, Python and Java (Android) interfaces are supported. It runs on many platforms such Windows and Linux and has more than 2500 optimized algorithms. It is widely adopted all around the world with more than 9 million downloads and still growing by nearly 200K/month [4]. It is extensively used for ranges of works from interactive art, to mines inspection, stitching maps, advanced robotics and manufacturing inspection. The sample applications to get started with the library are provided extensively and literally can be used as online tutorials for any application developer.

The hardware for running the OpenCV is full of choices as this library is already ported to many processor architectures and operating systems. One interesting hardware is a low-cost ARM based processor such as Raspbery pi, Cubie box, Odroid and alike with a variety of clock speeds, number of processing cores, advanced features and prices.

The Raspberry Pi 2 Model B is the second-generation Raspberry Pi [3] with more powerful features. It replaced the original Raspberry Pi 1 Model B+ in February 2015. Compared to the Raspberry Pi 1, it has a 900MHz quad-core ARM Cortex-A7 CPU with 1GB RAM. Provision of 4 USB ports, 40 GPIO pins, Full HDMI port, Ethernet port, Micro SD card slot and VideoCore IV 3D graphics core. These features make it a reasonable choice as an entry point to test, as there are other similar ARM miniPCs with higher specifications that have also been evaluated by the authors such as Odroid and Cubie Board families. These miniPCs run the full range of ARM GNU/Linux distributions, including Ubuntu Core. The price of this Raspberry Pi miniPC is lowest compared with other miniPCs from ARM family. It means there is a room for performance improvement if required later on.

Other hardware widely available in the market for low cost camera vision is a USB web camera. Webcam-c170 is a good choice with its features and requirement matched by general ARM miniPCs running at or above 1 GHz with minimum RAM of 512 MB. Interfacing is simply achieved with USB 2. Camera resolution of 640x480 pixels is a good start for video processing. For better performance, a faster miniPC is preferable together with higher and faster resolution of the web camera.

The authors have implemented earlier a real-time video processing system for robotic applications in a flexible manufacturing system successfully using OpenCV library and web camera with very promising results for real-time solution in 2010 [5]. For embedded system application using sound sensors [6] and other environmental sensors [7], the authors have also successfully developed multi sensors signal processing system that can be integrated into the human, environmental and energy monitoring system.

II. METHODOLOGY

The overall steps in generating a heatmap can be illustrated as in the following flowchart in Figure 1. The detail is as follows.

A. Preparing the development environment

Before starting the effort to generate the human vision heatmap to track the movement, the hardware and software have to be prepared and setup correctly, as mentioned in the part I. MiniPC with enough storage space on SD Card or SATA hard disk for running the Linux Operating System and software development tools such as Qt with OpenCV library has to be made ready with its keyboard, mouse, web camera and internet connection. Once the preparation is completed and the hardware/software runs cleanly, the next step is the video/image processing which is the key activity in this research.

B. Taking images

Interfacing and activating the web cam through USB 2 port and calling the device ID from within the OpenCV library allows us to access and control the web cam. Once the device is recognized, many variations of command to capture the live image from the web cam can be used to take the initial image. OpenCV library supports many syntaxes with a variety of options to get the intended actions.

It is also possible to reconfigure the web cam capture properties such as pixel resolution to match the processing capacity and memory availability of the miniPC. This feature offers flexibility and freedom to the application developers.

C. Clear the back image

Once the image capture is ready, the first reference image is taken and saved as the visual background image of the 'quiet' image with no 'object' in the image frame. cvAbsDiff is one of many commands that OpenCV library supports to calculate the absolute difference between two arrays when they have the same size and type. Passing the same image as an input image to the cvAbsDiff produces zero difference image which is a pure black image. This image serves as the base image for creating path and color gradation to be added in real-time into the original visual background image of the 'quiet' image.

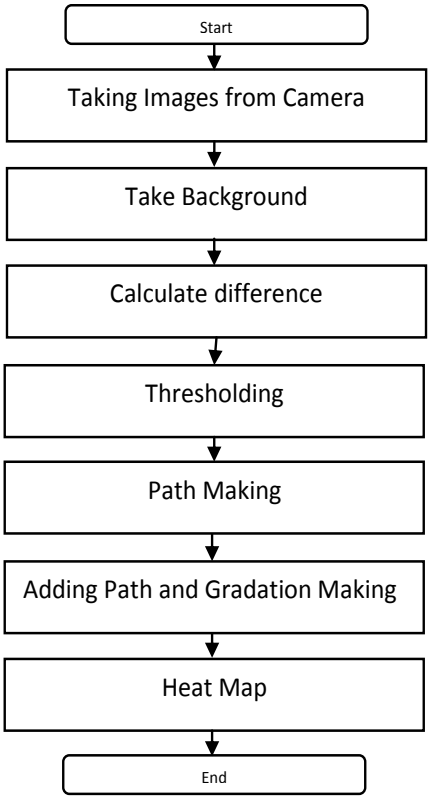


Figure 1: Flow chart of the system

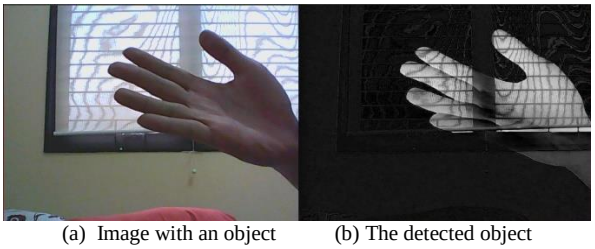
D. Calculate image difference / detecting movement

Using the same cvAbsDiff of OpenCV library is a clean way to detect movement. With cvAbsDiff a static background image will be compared with current video frames to obtain information about changes that happen at the current time, as illustrated in Figure 2. When no object is detected, the result of this operation is just a pure black image.

(a) actual background (b) zeroed background

Figure 2: Static background image.

In Figure 3, when there is an object, such as a hand entering the view port of the web cam, the operation will produce an object image of interest. Sometimes the object image still contains noise such as illustrated in Figure 3(b). This noise needs to be removed to improve the detection quality.

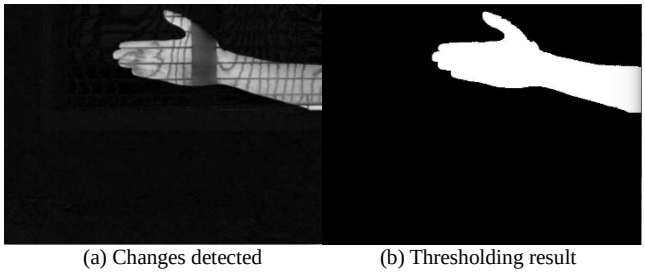


(a) Image with an object (b) The detected object

Figure 3. cvAbsDiff detects the hand as the changes on the background image

E. Thresholding to reduce/remove pixel noise

Thresholding is a method of image segmentation to transform a grayscale image to a binary image. Thresholding separates out regions of an image corresponding to objects of interest for further analysis. This separation is based on the variation of intensity between the object pixels and the background pixels, to produce a better image of pixel changes detected at that time. Figure 4 illustrates the operation.

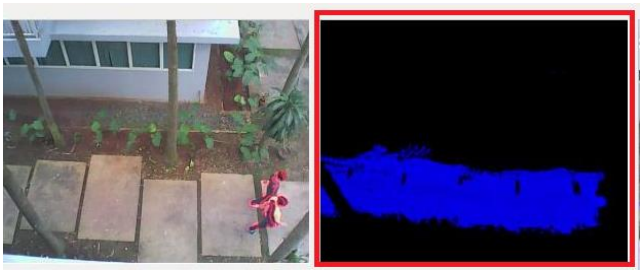


(a) Changes detected (b) Thresholding result

Figure 4. Detection and Threshold

F. Path-making of busy area

Using the thresholding result, a path is made by changing the RGB pixel values where the changes happen, as shown in the example in Figure 5. Two persons walking and passing the area under observation can be seen from Figure 5(a). The resultant path is shown in Figure 5(b) where the blue (or lighter color in gray scale image) is produced due to the detection.



a. Vision detects changes b. Accumulated pixel changes

Figure 5: Path creation based on changes in pixel values.

G. Path with color gradation

Color gradation is made to visualize human traffic in the specified area and in the specified time frame, such as in Figure 6. The RGB values of the path will be updated every time changes due to object movements happen along the path that is already generated earlier. This will create a different level of gradation depending on the detected activities in the observation space from time to time. Higher intensity path is shown here where the activity of human movement is high, while black area means there is no object detected.

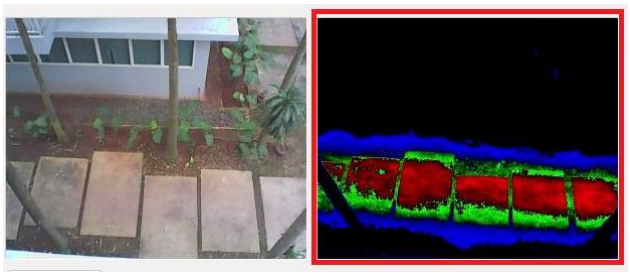


Figure 6: Gradation is produced when changes of pixel values occur along the same path.

H. Real-time vision heatmap overlaid on visual image

Adding the color gradation into the original static background image results in a clear visual picture that consists of color gradation. The resultant picture with color gradation is called heatmap, such as shown in Figure 7. From the image, it can be deduced that the path made of concrete is the path of the 'busiest' area, while the surrounding is 'quiet' area.

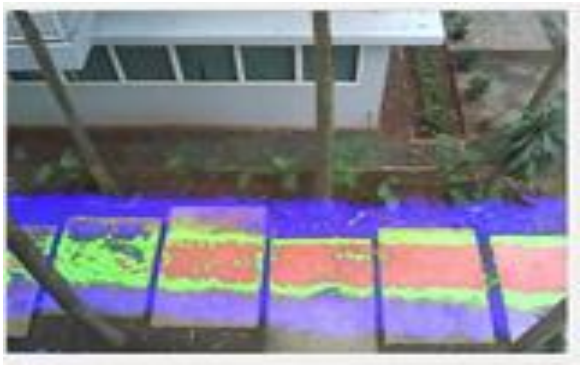


Figure 7: Heatmap

I. Image Stitching

Image stitching is also part of the OpenCV programming library. The purpose of stitching images in this research project is to produce a wider view of the final heatmap. The strategy is to combine two partially overlapping images into one image as can be seen in an example shown in Figure 8. Two partially overlapping images on the right side of Figure 8 are the source heatmaps with a long tree on the bottom right of the upper source image. Another long tree is on the bottom left of the bottom source heatmap. When both source heatmaps are stitched together, the result is a bigger heatmap on the left part of Figure 8. Here both long trees are reproduced on the final image, giving rise to wider observation areas. As many images as required can be stitched in this way, depending on the need and resource availability.

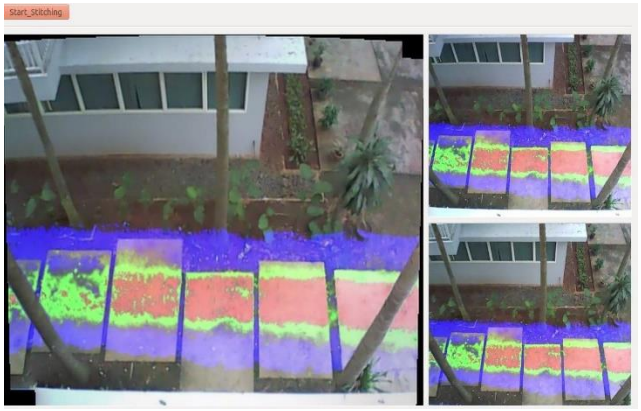


Figure 8: Image stitching result from 2 heatmaps

III. RESULTS AND DISCUSSION

Several tests have been performed to evaluate the overall process and performance of the vision heatmap system. The system is tested in 4 different lighting intensity. The first test was held outdoor daylight at 13:00 when the sun was so bright. The result from the first test is bad because of the light is too bright for the camera to capture the balanced image.

The heatmap result from the test at 13:00 cannot be seen well because of it is too bright and saturated to see as can be observed in Figure 9.

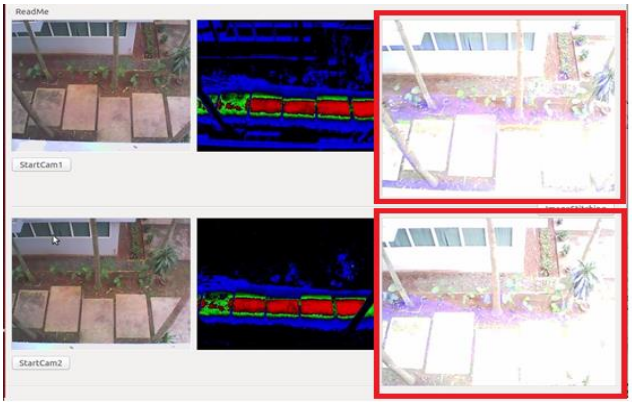


Figure 9: Test result at 13:00

The second test was held outdoor daylight at 15:00. The background image taken has good contrast and brightness. It is not too bright compared to the background taken at 13:00. The gradation of the color in the final image is well distinguishable. The final result of this heatmap test at 15:00 can be seen in Figure 10.

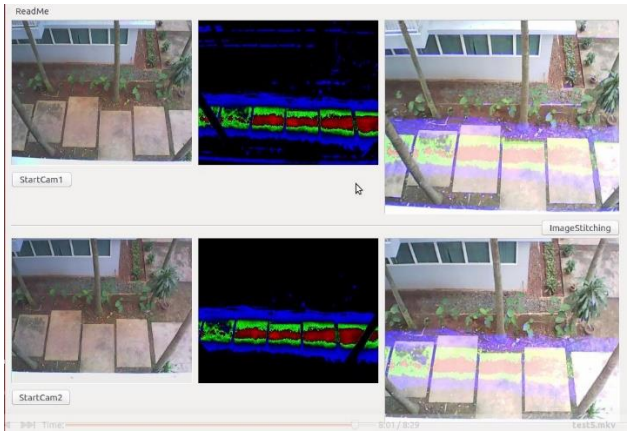


Figure 10: Test result at 15:00

The third test was held outdoor late daylight at 17:00. The background image taken from the test is still really good. The background image taken is the same as the actual area condition. This test gives the best heatmap result. The color gradation can be seen clearly in the result image. The final result at 17:00 can be observed in Figure 11.

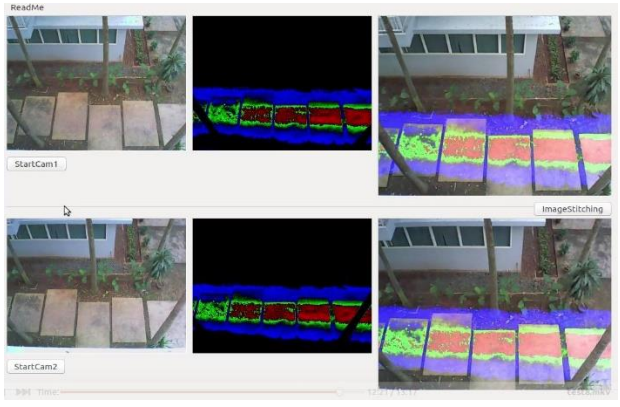


Figure 11: Test result at 17:00

The fourth test gives the worst result as it was done in the dark at 19.00 at the same location as the first 3 tests. The system cannot detect the human movement because the area is too dark. However, unintentionally in this test a person using a hand phone can be detected by the system. It is because the smart phone monitor emits light when it is activated. This light can be captured and recognized by the system.

The image that shows a person moving at the area but not detected by the system can be seen in Figure 12. The result is just pure black.

Figure 12: A walking man in the dark is not detected

The heatmap of a man using his smart phone detected by the system can be seen in Figure 13, as a moving point that creates an illusion of line on the dark background image. If an infrared camera was used, the background could easily be seen on the heatmap as well, thus provides an opportunity to use the heatmap system in the dark area.

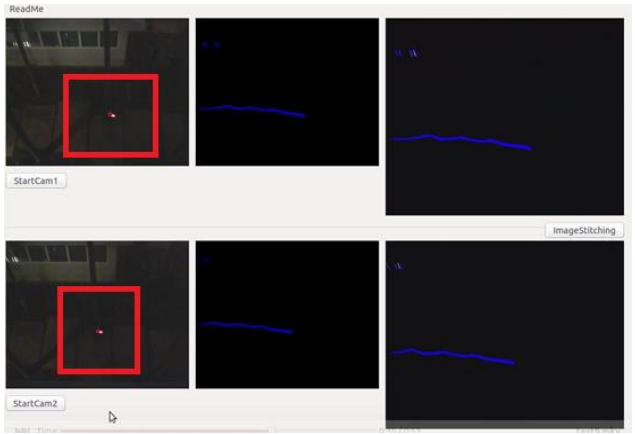


Figure 13: Light on a smart phone is detected by the system

All tests are recorded in the video to serve as a proof of systematic test and performance evaluation. Many other systems such as environmental and energy system can be integrated into heatmap system to produce a smart application for smarter building system.

IV. Conclusion

According to the objective of this research, designing and creating a low-cost system using the camera to detect movement based on differences between the background image and images taken in current time is successfully

achieved. Human movement can be tracked and recorded in a specified area and within specified time frame. The busiest and the quietest human movement behavior in a specified area and within the specified time frame can be defined by looking at the color gradation. Red (or lighter) color corresponds to the busiest area and blue (or darker) color indicates the quietest area of human movement behavior.

Using two cameras or more, a wider area can be accommodated and observed well, as heatmap stitching tests are successful. The heatmap system can work on the full potential on the specific lighting condition. The current heatmap system cannot work in the dark, but this limitation would be easily solved by utilizing infrared cameras instead of generic web camera to capture images in the dark. This type of camera is used extensively in security and surveillance system. The heatmap system process in Raspberry Pi 2 Model B is lagging due to its low CPU capability. This can be solved by selecting much higher performance ARM based miniPCs in the market.

Finally, the application can be extended not only for human behavior tracking but many other areas that use vision system, such as in energy efficient lighting system, rat tracking, manufacturing system. This technique can be integrated to create smarter systems.

ACKNOWLEDGMENT

The authors acknowledge the financial support in the form of research grant from the Indonesian ministry of Ristekdikti for the year 2016-2017, that enables the authors and team members to carry out research on the proposed topics of embedded system for diverse areas in general and smart renewable energy system in particular.

REFERENCES

[1] Heatmap, http://www.businessdictionary.com/definition/heatmap.html (last accessed Dec 14, 2016)
[2] OpenCV, https://github.com/opencv/opencv/wiki (last accessed Dec 14, 2016)
[3] Raspberry Pi, https://www.raspberrypi.org/products/raspberry-pi-2-model-b/ (last accessed Dec 14, 2016)
[4] Webcam, http://www.logitech.com/id-id/product/webcam-c170 (last accessed Dec 14, 2016)
[5] A. Djajadi, F. Laoda, R. Rusyadi, T. Prajogo, M. Sinaga, "A Model Vision on Sorting System Application Using Robotic Manipulator", TELKOMNIKA J., Vol. 8 No. 2, Augustus 2010, pp.137 – 148
[6] A. Djajadi, R. Rusyadi, T. Handoko, M. Sinaga, J. Grueneberg, "Analysis, Design and Implementation of an Embedded Realtime Sound Source Localization System Based on Beamforming Theory", TELKOMNIKA J., Vol. 7, No. 3, 2009 pp. 151-160.
[7] A. Djajadi, M. Wijanarko, "Ambient Environmental Quality Monitoring Using IoT Sensor Network", Internetworking Indonesia Journal Vol.8. No.1, 2016.