Real Time Multi Target Capturing Using Partitioning in Robot Vision

Davood Pour Yousefian Barfeh

College of Engineering and Computer Studies, Lyceum of the Philippines University-Laguna, Calamba, Philippines Department of Computer Science, Adamson University, Manila, Philippines Email: davoodpb@gmail.com

> Patrice Xandria Mari A. Delos Reyes Graduate School, University of the Philippines-Los Baños, Philippines Email: padelosreyes@up.edu.ph

> > Myrna A. Coliat

College of Informatics and Computing Sciences, Batangas State University, Batangas, Philippines Email: myrna_coliat@yahoo.com

Abstract— In this study, the authors design and implement a real time system as an autonomous robot-camera to capture many targets in the scene. The robot has only one camera. but it is capable of capturing more than one moving object through proper movement. The system uses Gaussian Filtering for motion detection and then performs partitioning to grab location of all targets in the scene. Due to partitioning, the scene has three major regions while each of which has different sub-regions. Based on the partitioning and position of all targets, the system might be in three states of unsafe state, safe state, and over-safe state. In each state regarding specific regions or sub-regions, the system picks appropriate movement not only to be capable of capturing all moving objects, but also to give equal chance of capturing to new targets entering to the scene from different direction. The system is tested in both of indoor and outdoor with different values for different parameters such as resolutions, fps (frame-per-second), minimum number of motion frames, and minimum areas of motion.

Index Terms— Motion detection, Gaussian filtering, multi target tracking, moving objects, partitioning, digital image processing

I. INTRODUCTION

There are three different levels of algorithms in the field of digital image processing; low level, intermediate level, and high level vision algorithms [1][2]. In low level vision algorithms, both of the input and output are digital images same as converting an ordinary colored image to a grayscale image or even to a binary image. In intermediate level vision algorithms, the inputs are digital images, but the outputs are symbolic description of images' features same as contours detection in an image. In high level vision algorithms, both of the input and output are symbolic representation of images' features [1,2]. As the names of these three levels show, high level vision algorithms are more difficult and more important, since they are dealing with perception and the context. Because of being context based, high-level vision

algorithms play vital roles in object recognition and image understanding. The best examples of high level vision algorithms are motion detection or tracking in which the goal is to detect objects which are moving [1] [2].

There are many challenges and complications in motion detection and tracking such as changes in illumination, dynamic background, occlusion, clutter, camouflage, presence of shadow, motion of camera, noise of video, challenging weather, speed of the moving objects and intermittent object motion [3]. Despite of these problems, motion detection and tracking are fascinating arenas in computer vision and digital image processing, since they are bases in many fields such as supplying advanced interfaces between humans or between humans and devices, military, medicine, sports, and specifically smart surveillance cameras or PTZ (Pan-Tilt-Zoom) cameras [3–5].

II. RELATED STUDIES AND PRODUCTS

With a simple search many studies or products can be found in the field of target (moving object) tracking and capturing. All of these researches and products can be categorized into four groups.

Group-1 contains researches and studies which are not performed based on camera, but researchers try to detect and track many moving objects in videos [6-12]. Here, the term "track" means that path of moving objects. In this group, the systems do not work real-time and in result, the systems do not pay attention to the motion of camera.

Group-2 contains researches and studies that work real-time. Here, the researchers focus on detecting and tracking of many moving objects using stationary cameras [13–15]. In these studies same as studies in group-1, the term "track" means that path of moving objects. It means these studies try to find the paths of moving objects. Moreover, in these researches, motion of camera is not considered, since the studies are based on fixed cameras.

Group-3 consists of researches performed based on a moving camera. Here, the researchers consider detecting and tracking of many moving objects while the camera has an unknown (random) motion [16–18]. In group-3 same as the studies in group-1 and group-2, the term "track" means that path of moving objects. In addition, the motion of camera is not dependent on motion of moving objects.

Group-4 contains industrial products and the studies in which the researchers used non-stationary cameras to detect and track moving object [19–21]. Cameras in this group are capable not only to detect the moving object, but also to track and capture the moving object by motion or by focal zooming. Products and systems of this group are able to capture and follow only one target.

III. STATEMENT OF THE PROBLEM

In reality, when we talk about motion and detection of motion, most of the time there are many objects which are moving – not only one object. Therefore, we need a multi-target capturing system that can capture many moving objects. Moreover, the system must be capable of moving to continue capturing of targets, since targets move and their movements might be in different directions.

As explained earlier, there exist numerous studies, systems and potent products regarding multi-target detection and tracking, but none of them is suitable for multi-target capturing, since studies in group-1 detect and track motions in a captured video. These studies do not care about capturing and following of targets, but detect targets and then find paths of the targets. Studies in group-2 can find track of targets. In fact, the focus of studies in group-2 is not capturing and following of targets that is why in these studies, systems cannot move. Studies in group-3 are able to find paths of targets while camera has some motion. In fact, studies in group-3 are more intelligent than studies in group-2, they can find paths of targets while system has a random movement. Although the systems in group-3 have movement, the movement is not for capturing the targets, but an input parameter. Finally, systems and products in group-4 are able to perform target capturing and following, but only over one target. If there are many targets in the scene, they select one of the targets, then focus on the target and then follow and capture it. In fact, studies in group-4 pay attention to only one target, and leave other targets.

After considering all groups of studies and products, the researchers observed that there is no algorithm or system to capture many moving objects (targets).

IV. PROPOSED SYSTEM

To solve the problem mentioned, the researchers propose Real Time Multi-Target Capturing using Partitioning in Robot Vision (RTMTCPRV). The system RTMTCPRV is a robot having a camera that not only detects multiple moving objects, but also can move forward, backward, left or right to continue capturing of the moving objects.

Specifically in this study, researchers design and implement an algorithm that;

1) Performs multiple moving objects detection (motions detection), and

2) Tries to maintain capturing of the moving objects.

For each of which, the system needs some algorithm. First, the system uses an algorithm to find all targets in the scene and then uses some algorithm to move (if needed) such that to be able to continue capturing of the targets.

A. Motions Detection

There are many approaches for motion detection in a continuous video stream. All of them are based on comparing of the current video frame with one frame (or more) from the previous frame (or frames) that is called as background [22]. In particular, this study uses Gaussian filtering for motions detection. In digital image processing, Gaussian filtering (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function that is widely used in graphics tools to reduce noise of image or even reduce some details in image. The influence of the Gaussian filtering is producing a smooth blur resembling image [22–26].

B. Movement

When the scene is captured and some motions are detected, then the system may need to move. The system tries to follow all of the moving objects using movement. In other words, the system may move forward, backward, right or left to maintain capturing all targets in the scene. In this regard, the scene is parted into three major partitions; unsafe area, safe area and over-safe area as shown in Fig. 1.



Figure 1. Three major partitions

There are two important areas inside the safe area; right side and left side. Moreover each area in safe area (right side and left side) has two parts. Totally, safe area has four important sub-areas; sLL (safe, left side of left side), sLR (safe, right side of left side), sRR (safe, right side of right side) and sRL (safe, left side of right side. There are also four important sub-areas inside unsafe area; uLL (unsafe, left side of left side), uLR (unsafe, right side of left side), uRR (unsafe, right side of right side) and uRL (unsafe, left side of right side) as shown in Fig. 2.



Figure 2. Sub-areas inside safe area and unsafe area

All movements of the system are based on two rules regarding their priorities;

Rule-1. Keep the targets in the safe area, and

Rule-2. Keep the targets in the center (not left and not right).

Rule-1 that is always prior to Rule-2, is equivalent to two sub-rules regarding their priorities;

a) Not to keep any target in unsafe area for not losing the target, and

b) Not to keep any target in over-safe area for capturing the target better, having more details.

When some target is inside unsafe area, there is a risk that the target gets out of scene especially when the target is moving fast. On the other hand, when all targets are inside over-safe area, system can get closer to them and can capture them with more details.

Rule-2 says that if there is no target in unsafe area and all targets are inside only one of the four sub-areas of safe area, then the system moves such that all targets are placed inside the center of the scene. We use Rule-2 to give equal chance of capturing to all targets that will come inside the scene from different directions.

Based on partitioning and position of the targets, the system can be in three states; unsafe sate, safe state and over-safe state.

Unsafe state is the case in which there exists at least one target in unsafe area. In this case, system has to move backward, right or left to follow the Rule-1, part a. In fact, system decreases the risk of losing the targets via not including targets in unsafe area.

Safe state is the case in which there is no target (there is no motion in the scene) or there is at least one target in safe area while there is no target in unsafe area. System is said to be in SSNM (safe state and no move) when;

- a) The system is in safe state and there exist at least one target in the left side of safe area and at least one target in the right side of safe area, or
- b) The system is in the safe state and there is no target in the right side of safe area and there is no target in the left side of safe area.

When the system is in SSNM, it does not move. System is said to be in SSM (safe state with move) when the system is not in SSNM. When the system is in SSM, it moves toward occupied side (left or right) to follow Rule-2.

The system is said to be in over-safe state when the system is neither in unsafe sate nor in safe state. In this case, system moves forward to follow the Rule-1, part b. In fact, system increases the details of capturing via getting closer to the targets.

When system starts, the variable target is false, since there is no detected target yet. The function motion_detection detects all targets among all areas in the scene. This function changes the variable target to true if there is a motion. This function also returns the areas of all targets. Then the system realizes its state based on the areas of the targets.

If the system is in unsafe state, then follows Rule-1 (part a) and changes the state to the safe state using move_right, move_left, or move_backward.

If the system is in SSM, then follows Rule-2 using move_right or move_left. It is noticeable when the system is in SSM, there is no need to pay attention to Rule-1, since the system is neither in unsafe state nor in over-safe state. In other words, both part of Rule-1 (part a and part b) are already followed.

If there is some motion – the value of variable target is true – and the system is neither in unsafe state nor in safe state, then the system is surely in over-safe state. In this case, system follows Rule-1 (part b) using move_forward to change the state from over-safe state to the safe state.

C. Unlimited Backward

If the system is in SSM, then follows Rule-2 using move_right or move_left. It is noticeable when the system is in SSM, there is no need to pay attention to Rule-1, since the system is neither in unsafe state nor in over-safe state. In other words, both part of Rule-1 (part a and part b) are already followed.

As mentioned earlier, generally motion detection is based on comparing of the current video frame with one frame (or more) from the previous frame (or frames) that is called as background. In other words, when there is some difference in some pixels in two (or more) consequent frames, then the system considers the difference as a motion. In other hand, when system moves, almost the entire scene changes and in result pixels of almost all areas of consequent frames change. In this case, system adjudges to have motions in all areas including sub-areas of unsafe area. Here, system proceeds to unsafe state and has to perform move_backward to change the state to safe state while the result of movement is unsafe state again. It means that, with first movement, the system falls in an unlimited loop of unsafe state and then move_backward.

To eliminate this problem, the simple idea is that the system should not perform motions detection during its movement.

D. Algorithm

To eliminate this problem, the simple idea is that the system should not perform motions detection during its movement.

The overall algorithm for the system RTMTCPRV is in the following.

target = false

motion_detection()
while (target == true){

if (unsafe_state == true){

if (uLL == true AND sRR == false AND sRL == false)

{ move_left; move_left; stop motions detection; } elseif (uRR == true AND sLL == false AND sLR == false) { move right; move right; stop motions detection; } elseif (uLR == true AND sRR == false) { move_left; stop motions detection; } elseif (uRL == true AND sLL == false) { move right: stop motions detection; } else { move backward: stop motions detection; } exit: } // end of if unsafe state if (safe state == true){ if ((sRR == true OR sRL == true) AND sLL == false AND sLR == false){ move right: stop motions detection; } elseif ((sLL == true OR sLR == true) AND sRR == false AND sRL == false) { move left: stop motions detection; } exit: } // end of if safe state move forward: // over-safe sate stop motions detection; } // end of while

When the robot starts, no moving object has been detected that is why the value of the variable target is false. The function motion_detection has responsibility of detecting all targets (motions) and then changing the value of the variable target to true in the case of detecting motion. Then regarding the state and particular areas and sub-areas based on position of the target, the robot moves to keep capturing the target s. As seen in algorithm, when the robot is moving, the function motion_detection stops because of the danger of unlimited backward.

E. Implementation

The system RTMTCPRV was implemented as a robot camera consisting a Raspberry Pi 3 Model B with a camera module v1, two TowerPro SG5010 servo motors each of which connected to a wheel, and power supplies. Researchers used Raspbian Jessie as operating system, OpenCV as library, and Python as programming language.

To implement the algorithm into the actual system, there are four major concepts which need to be explained in details. These concepts consist of partitioning, motions detection, movement unit, and stop motion detection.

1) Partitioning

The system uses the width and the height of the image (frame) to divide the scene to different areas and subareas. Each frame of the scene is divided to many regions vertically based on the height of the frame and horizontally based on the width of the frame. Desirable areas and sub-areas can be achieved using intersection (AND) and union (OR) of the regions vertically and horizontally.

2) Motions detection

As mentioned earlier in 4.1, the system uses Gaussian filtering to detect all targets in the scene. The researchers used the function "GaussianBlur", for Gaussian filtering. Afterwards, functions "accumulateWeighted", and "absdiff" are used to accumulate the weighted average

between the current frame and previous frames due to compute the difference between the current frame and running average. Then the functions "threshold" and "dilate" are used to threshold the delta image and dilate the thresholded image to fill the holes in the image. Finally the functions "findContours" and "contourArea" are used to find area of each motion.

The actual values of the parameters used in the functions mentioned above include; width and height of kernel size = 21, Gaussian kernel standard deviation in X direction and Y direction = 0, weight of input image = 0.5, threshold value = 5, maximum value of thresholding = 255, thresholding type = THRESH_BINARY, size of structuring element in dilation = 21, contour retrieval mode = RETR_EXTERNAL, and contour approximation method = CHAIN_APPROX_SIMPLE.

3) Movement unit

After detecting targets in the scene, system may need to move to right, to left, forward or backward. Since the system does not have any movement in vertical axis, the unit of movement is based on horizontal angle of view of the camera. As Figures 1 and 2 show, the scene is basically partitioned into a 10×10 -unit page, while horizontal angle of view of the camera is 53.50 [27]. Hence, the movement unit is equal to 5.35 degrees. Generally the formula of horizontal movement unit is:

$$HMU = HAV/HBP \tag{1}$$

Where HMU represents horizontal movement unit, HAV represents the horizontal angle of view, and HBP represents the horizontally basic partitioning of the scene.

4) Stop motion detection

When system is moving, it has to stop detecting motions for not going inside the unlimited backward loop. To perform this, the function "time.sleep(t)" is used to make a delay in which from software view, the system performs nothing. To compute the parameter "t" that represents the desirable delay time in second, speed of the servo motor is used as the basis that is equal to 0.19 seconds/60 degrees, when the input voltage is 4.8 volts [28] [29]. This speed means that 0.19 seconds are needed to move 60 degrees that is equivalent to 0.00317 seconds per degree. Hence, the time needed to have 5.35 degrees (movement unit) is about 17 milliseconds. In other words, each movement of system lasts for 17 milliseconds. Thereupon, the parameter t is equal to 0.017.

V. RESULT AND DISCUSSION

Researchers tested the system RTMTCPRV in outdoor and indoor with different resolutions, fps (frame-persecond), minimum number of motion frames, and minimum areas of motion.

In indoor whilst the lights are on and the maximum number of moving objects is 6 – containing electrical fans, dogs and humans while their distances from system are in the range of 0.5 to 5 meters, the best results are obtained when the resolution is at least set to 640×480 , fps is at least set to 16, minimum number of motion frames is set to 8, and minimum area of motion is set to 1500 pixels.

In outdoor - an open parking area, the story is complicated. In fact, the problem in outdoor is related to the motions detection in wide range of distance. In this case if the minimum area of motion is set to a low value (for example, 500) to detect targets while they are far more than 9 meters, then the system detects many illusory targets. On the other side, if the minimum area is not set to a low value, then the targets more than 9 meters far cannot be detected. Aside from the problem, if the maximum number of moving objects is 10 - containing cars, motorcycles, dogs and humans while their distances from system are in the range of 5 to 9 meters, the best results are obtained when the resolution is at least 1296×972, fsp is between 16 and 42, minimum number of motion frames is 8, and minimum area of motion is 1500.

REFERENCES

- P. Y. Barfeh, D. Balba, N. Bustamante, R. Jose, E. Albacea, and V. Mariano, 2017. "Multi-informational auto tracking algorithm," in Proc. 2nd International Conference on Information in Business and Technology Management (I2BM 2017).
- [2] L. Pitas, Digital Image Processing Algorithms and Applications, 2000.
- [3] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern Recognition*, vol. 36, no. 3, 2003.
- [4] H. Lina, I. Shafranb, D. Yuhc, and G. Hagera, "Towards automatic skill evaluation: Detection and segmentation of robotassisted surgical motions," *Taylor&francisOnline*, pp. 220-230, 2010.
- [5] A. Caballeroa, J. Castilloa, J. Cantosc, and R. Tomasd, "Optical flow or image subtraction in human detection from infrared camera on mobile robot," *Robotics and Autonomous Systems*, vol. 58, no. 12, pp. 1273–1281, 2010.
- [6] W. Choi and S. Savarese, "A unified framework for multi-target tracking and collective activity recognition," in *Proc. 12th European Conference on Computer Vision, Florence*, Italy, 2012.
- [7] A. Butt and R. Collins, "Multi-target tracking by lagrangian relaxation to min-cost network flow," in *Proc. IEEE Conference* on Computer Vision and Pattern Recognition, 2013.
- [8] A. Andriyenko, K. Schindler, and S. Roth, "Discrete-Continuous Optimization for Multi-Target Tracking," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
- [9] Z. Qin and C. Shelton, "Improving multi-target tracking via social grouping," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
- [10] B. Yang, C. Huang, and R. Nevatia, "Learning Affinities and Dependencies for Multi-Target Tracking using a CRF Model," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- [11] A. Andriyenko and K. Schindler, "Multi-target tracking by continuous energy minimization," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- [12] B. Benfold and I. Reidstable, "Multi-target tracking in real-time surveillance video," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- [13] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, "Fast and robust hand tracking using detection-guided optimization," in *Proc. Conference on Computer Vision and Pattern Recognition*, 2015.
- [14] S. Melax, L. Keselman, and S. Orsten, "Dynamics based 3d skeletal hand tracking," *Canadian Information Processing Society*, 2103.
- [15] J. Xing, H. Ai, and S. Lao, "Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses," in *Proc. IEEE Conference* on Computer Vision and Pattern Recognition, 2009.
- [16] B. Jung and G. Sukhatme, "Real-time motion tracking from a mobile robot," *International Journal of Social Robotics*, vol. 2, no. 1, pp. 63–78, 2010.

- [17] W. Choi, C. Pantofaru, and S. Savarese, "Detecting and tracking people using an RGB-D camera via multiple detector fusion," in *Proc. International Workshops on Computer Vision*, Daejeon, Korea, November 5-6, 2012.
- [18] B. Yang and R. Nevatia, "Multi-target tracking by online learning of non-linear motion patterns and robust appearance models," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
- [19] Axis. [Online]. Available: http://www.axis.com/files/datasheet/ds_215ptz_34462_en_0902_1 o.pdf (Accessed August 22, 2017).
- [20] Bhphotovideo. [Online]. Available: https://www.bhphotovideo.com/c/product/752624-REG/Samsung_SPU_3750T_1_4_Vertical_Double_Density.html (Accessed August 22, 2017).
- [21] Bkgrupe. [Online]. Available: http://www.bkgrupe.lt/uploads/docs/Hikvision/DS-2DF8836IV-AELW.pdf (Accessed August 22, 2017).
- [22] L. Shapiro and G. Stockman, *Computer Vision*, 2001.
- [23] M. Nixon and A. Aguado, *Feature Extraction and Image*
- Processing, Academic Press, 2008.
 [24] K. Ahi, "Modeling of terahertz images based on x-ray images: a novel approach for verification of terahertz images and identification of objects with fine details beyond terahertz resolution," in Proc. SPIE 9856, Terahertz Physics, Devices, and Systems X: Advanced Applications in Industry and Defense, 2016.
- [25] E. Reinhard, High dynamic range imaging: Acquisition, display, and image-based lighting, Morgan Kaufmann, 2006.
- [26] R. Fisher, S. Perkins, A. Walker, and E. Wolfart, Spatial Filters -Laplacian of Gaussian, 2003.
- [27] Raspberrypi. [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/ (Accessed August 22, 2017).
- [28] Towerpro. [Online]. Available: http://www.towerpro.com.tw/product/sg5010-4/ (Accessed August 22, 2017).
- [29] Servodatabase. [Online]. Available: https://servodatabase.com/servo/towerpro/sg-5010 (Accessed August 22, 2017



Davood Pour Yousefian Barfeh is a Persian (Iranian) researcher who was born in Iran. He studied BS Computer Engineering and BS Computer Science. He also studied MS IT and MS Computer Science. He finished his PhD Computer Science in 2018. His fields of interest are; Digital Image Processing, Robotics and Theoretical Algorithms.



Patrice Xandria Mari A. Delos Reyes is a Filipino researcher. She finished BS in Biology and MS in Development Communication with a minor in Computer Science. She is a member of the UPLB Society of Pre-Med Students and the Philippine Society of Microbiology Inc. She is currently studying PhD in Development Studies with minors in Environmental Science and

Strategic Planning and Policy Studies.



Myrna A. Coliat is an Associate Professor at Batangas State University, where she is the Director for Institutional Planning and Development. She obtained her Master's degree in Computer Science from De La Salle University, Manila, Philippines. She is presently pursuing Doctor of Technology at Batangas State University.