Multiscale Visual Object Detection for Unsupervised Ubiquitous Projection Based on a Portable Projector-Camera System

Thitirat Siriborvornratanakul and Masanori Sugimoto
Interaction Technology Laboratory
Department of Electrical Engineering and Information Systems, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, JAPAN
e-mail: { thitirat, sugi}@itl.t.u-tokyo.ac.jp

***Abstract*—Ubiquitous projection is the recent effort that tries to close the gap between the physical world and the virtual world by using a mobile projector. Using a projector and camera together has a conflict of preferred light conditions, making it difficult to implement a robust visual detector while retaining ubiquity. In this paper, we focus on techniques of visual object detection for a portable projector-camera system. The goal is to create a visual detector that requires no guidance from a user and is robust to different light conditions. Our investigation involves the multiscale concept using Canny edge detection as a representative detector. Five image simplification filters applied to the multiscale detection are examined for both accuracy and speed. In addition, preprocessing using histogram equalization and postprocessing are applied to ensure robustness in a real-world scenario, and to guarantee that the detection will always successfully detect objects using a constant set of parameters defined offline. Finally, we showed that using multiscale detection in a parallel manner can speed up the detection while not affecting the accuracy of the detection.**

I. INTRODUCTION

Because of the trend in mobile projector phones, people have started to imagine using mobile projectors ubiquitously in the same way as mobile phones. A good ubiquitous system is a matter of mobility and intelligence. While the mobility of a projector has recently been successfully proposed, intelligent mobile projection has not yet been accomplished in practice. In this paper, we focus on a vision-based ubiquitous projection system that is able to perform real-time object augmentation in a robust manner.

Object augmentation using a projector has been proposed continuously in the past decade; for example, see iLamps [7], Cao and Balakrishnan [1], CoGAME [2], Molyneaux et al. [5], and Kanbara et al. [3]. This trend has become more prominent with the proposal of the SixthSense project [4] that uses a wearable projector to augment the physical world with virtual information projected from a projector. There is no doubt that the key success of object augmentation is to detect the object correctly. This may sound simple, but it is not easy to achieve, as ubiquitous projection implies unconstrained environments and dynamic objects.

Previous works of augmenting objects by projection tend to neglect or simplify the detection problem with various alternatives that do not meet the goal of being ubiquitous. Visual fiducials and IR LEDs are attached to the target object in iLamps [7] and CoGAME [2], respectively, for visual object detection. Invisible markers need to be projected steadily onto a wall or ceiling in [3] and a stationary camera is required in the workspace for 3D position tracking in [1]. In other words, these works are not automatic and ubiquitous yet, because they either modify external appearances of the object or require at least one stationary device. In SixthSense [4], color markers are used for some object detection tasks but it is not clear how the visual object detection is performed in the rest of the system. The cooperative augmentation proposed by [5] seems to be the appropriate solution for real-time object augmentation in ubiquitous projection. The projector-camera (i.e., pro-cam) system dynamically configures its visual object detection based on four different detection algorithms that ensure detection coverage in a real-world scenario. However, this system relies on an assumption of the smart object where object-model knowledge must be embedded during manufacture.

The pro-cam system using visual object detection is often the choice for researchers in this field. It allows object augmentation everywhere without relying on separate tracking hardware, and a user can interact directly with the projection. Nevertheless, there is a conflict using projection and visual detection together. Projection requires a dark environment for better visualization of the projected images on the surface. In contrast, the visual object detection algorithm usually prefers input images captured in a bright environment so that the sharp details of objects are obtained with few additive noises. Combined with the fact that visual appearances of the object can be changed easily by the surrounding light conditions, the problem of visual object detection becomes more challenging for a ubiquitous pro-cam system.

So far, researchers in the field of ubiquitous pro-cam systems focus on developing new interactive techniques but pay scant attention to the fundamental problem of how to appropriately deal with the difficulties of lighting in a real-world scenario. In this paper, we closely investigate visual object detection using a ubiquitous pro-cam system. The goal is to accomplish an unsupervised visual object detection scheme that retains absolute ubiquity in a pro-cam setup and is able to reliably detect objects inside the projector's

978-0-7695-4271-3/10 $26.00 © 2010 IEEE
DOI 10.1109/DICTA.2010.109

frustum, regardless of extremely low light condition or significantly changing the amount of light. Contributions of this paper can be applied to other vision-based pro-cam systems to ensure robust object detection that tolerates surrounding light conditions. By simply replacing the representative detector used in this paper (Canny edge detector) with an appropriate visual detector, various object detections can be achieved for further use in object augmentation. We strongly recommend a reader to look at the digital copy of this paper because some fine details cannot be seen when printed onto a paper.

II. Pro-cam Design for Geometry-based Ubiquitous Interactions

This section briefly describes our main focus, which is the beam splitter-based scenario described in [8], where the fundamental lighting difficulties mentioned above are significantly underlined. In the design proposed by [8], available light as seen by the camera is very limited for two reasons. First, the beam splitter used to coaxialize the projector and camera allows only 50% of the environment light to be reflected to the camera lens. Second, the design refers to pro-cam synchronization where the camera is set to expose for a very short period (i.e., less than 1 ms). Consequently, the projector's red light is the only light source illuminating the projection surface, and small changes in distance from the projector to the object, or slight depth variation, can result in considerably increasing or decreasing the amount of light as seen by the camera.

Advantages of this design are specifically for geometry-based interactive projection in a ubiquitous system. Not only is the geometric mapping between the projector and camera coordinates independent from the surface because of the coaxialization, but also the projected augmentation will not be seen by the camera. In other words, this design allows the use of conventional visual object detection, and the detection and projection can be performed simultaneously. As shown in Fig. 1, the design in [8] can detect the true object correctly (Fig. 1D and 1G) while a normally set camera cannot (Fig. 1C and 1F). The limitation of this design is that the projected colors must be converted to a specific set of colors. This can be seen in Fig. 1A where all projected colors are limited to the red highlights.

All input images used in the rest of this paper are captured by this pro-cam design, as this best emphasizes the problems of a ubiquitous pro-cam device.

III. Multiscale Visual Object Detection

In this section, we explain our investigation for unsupervised visual object detection. Unlike color and texture, edges generally present across any type of visual content. Hence, Canny edge detection provided by the OpenCV library [6] is used as a representative base detector in this paper; λ_1, λ_2, and σ represent values of two Canny thresholds and aperture size, respectively. Note that all input images used in this paper are unsigned 8-bit RGB images.

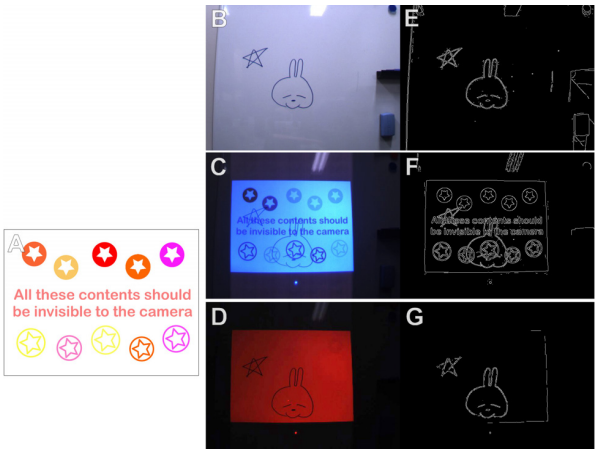

Figure 1. (B) An environment seen by the normal camera. (C) and (D) show environment (B) as seen by the normal camera and the synchronized camera (according to [8]), respectively, while (A) is being projected from the projector. (E), (F) and (G) are Canny edge detection results of (B), (C) and (D), respectively.

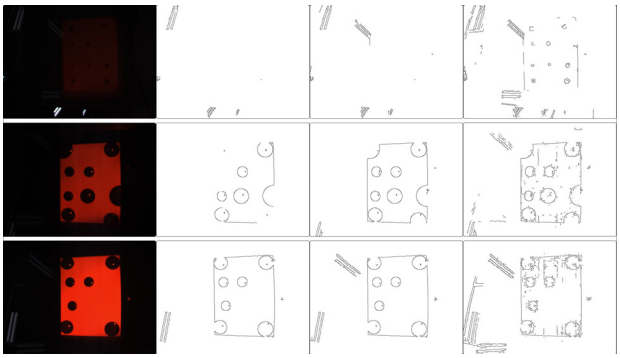

Figure 2. Comparison of visual object detection of three images with different illuminations. The first column shows images captured by the synchronized camera (according to [8]). The second, third and fourth columns are Canny edge detection results using $(\lambda_1, \lambda_2, \sigma)$ equal to (127, 1000, 5), (127, 500, 5) and (127, 200, 5), respectively.

For ideal visual object detection, one would like to extract automatically all edges belonging to the true objects. However, in an unconstrained environment, no matter what detector is used, its output tends to miss some true objects (a.k.a. undersegmentation) or include false edges belonging to noises (a.k.a. oversegmentation). Considering a single input image, it is always possible to achieve an acceptable detection result by fine-tuning parameters of the detector. An example is shown in Fig. 2, where the appropriate set of parameters is different in the three input images. The darkest image needs the parameters providing highly sensitive edge detection to extract low-contrast edges. However, using the same set of parameters in the brighter images produces oversegmented results.

Adjusting these parameters manually during real-time object augmentation is not preferred. Therefore, we closely investigated this problem and conducted experiments to

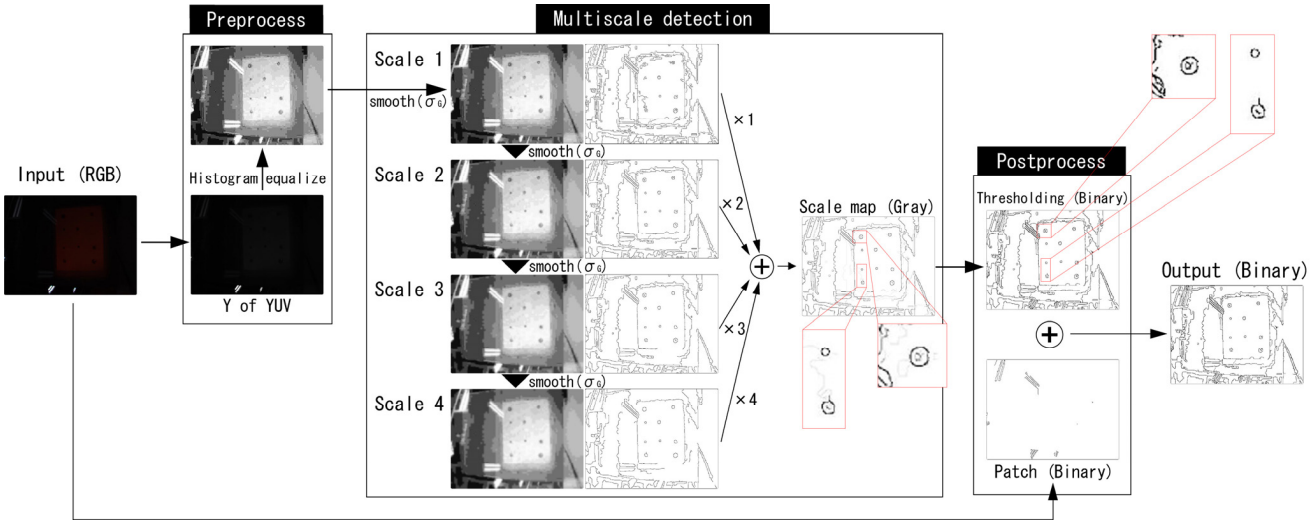

Figure 3. Overview of the investigated multiscale visual object detection. The Gaussian smooth filter is used in this figure.

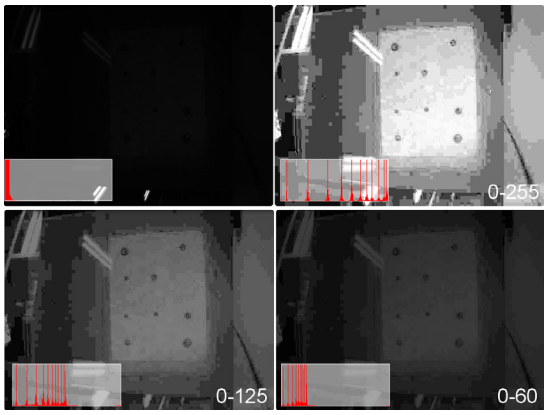

Figure 4. Effect of histogram equalization. Top-left is the original luminance component of the input image. The others are the luminance image whose histogram is equalized with a different range of spectrum (written in white digits). Corresponding histograms are drawn in red at the bottom-left of each image.

ensure that acceptable detection results will always be achieved with a single set of predefined parameters, regardless of different light conditions. After some surveys, we decided to use the multiscale concept to compensate the oversegmented result with the undersegmented result and vice versa. The reasons that we chose this concept are as follows: (1) it is easy to implement because no internal modification is required in the base detector, (2) it is flexible because the base detector can be replaced easily with other visual detectors to match individual requirements, and (3) increases in computation can be offset by the recent growth of multicore processors and parallel programming languages (as described later in Section III.D).

An overview of the investigated detection scheme is shown in Fig. 3. The detection involves three main steps: preprocess, multiscale detection, and postprocess. Section III.A explains the preprocess step, including our experimental results on the appropriate histogram equalization value. Section III.B shows in detail how we obtain the scale map from the multiscale concept (using the conventional sequential implementation), and how different smoothing filters affect the detection results. Section III.C then describes the postprocess step that converts the gray scale-map to the binary edge-map and fixes some special cases. Finally, Section III.D presents an alternative implementation using parallel programming to improve performance. The investigated detection scheme is automatic and involves no user feedback, training or supervision during online execution.

A. Preprocess: Histogram equalization

The purpose of this step is to standardize the input image so that it is ready for the next detection step. From our experiments, using only the multiscale concept is not enough to achieve an unsupervised detection; adjusting parameters of the base detector is still required when dealing with differently illuminated input images. Hence, we concluded that the input images need to be standardized first. The investigated preprocess includes extracting the luminance component of the input image and performing histogram equalization.

Instead of using the RGB format, we convert the image to YUV format and use only the Y (luminance) component in further calculations. This is related to the observation that human eyes are far more sensitive to luminance than color. Because our input images (as well as other pro-cam systems operating in a dark environment) suffer from insufficient amounts of light, histogram equalization is a straightforward technique to improve the overall appearance of the images. Histogram equalization is a technique that enhances an image by equalizing its histogram, and is usually used to enhance contrast of an image. Generally, this term refers to expanding the pixel values within an image to fill the entire 0–255 spectrum in its histogram.

As shown in the top row of Fig. 4, the original histogram of the luminance image occupies narrow ranges in the entire spectrum and equalizing it obviously improves the visualization. However, there is an issue caused by the increase of textural noises. The more the histogram is changed, the more distinct textural noises become, which may affect future visual detection. The textural noise in this context refers to the texture that does not really exist but is added by using the histogram equalization to strongly enhance the image. An example of textural noises can be seen in the equalized image of range 0–255 in Fig. 4. Decreasing the strength of enhancement reduces visibility of the textural noises, as shown in the bottom row of Fig. 4. Nevertheless, because luminance of the input image is unpredictable, we decided to equalize the histogram to cover the entire spectrum (i.e., 0–255) as the standard. Effects of textural noises will be resolved later in the multiscale detection step.

B. Multiscale detection: Smoothing filters

In this context, multiscale detection refers to a hierarchy of detections where an original image is simplified through iterations of a smooth filter as illustrated in Fig. 3 (σ_G is the size of the square Gaussian kernel used in the Gaussian smooth filter). The idea is to use a smooth filter (with constant parameters) to decrease iteratively "too detailed" information from the original image so that only the information belonging to the true objects survives in the last scale. By applying the base detector with the constant set of parameters to the simplified image in each scale, we can combine all detection results to a single output using a weighted summation. Suppose that N is the number of scales and i is an index of each scale, i equal to 1 refers to the highest scale (maximum detail but least importance), i equal to N refers to the lowest scale (minimum detail but highest importance), and the weight assigned to each scale is equal to i in this paper. In this way, the high magnitude (black) in the combined gray scale-map reflects the high probability that this pixel belongs to the true edges.

Nevertheless, as mentioned in Section III.A, significantly equalizing the histogram adds distinct textural noises to some images and can result in false positive visual detection. Using an appropriate smooth filter to simplify the image is important to ensure that these textural noises are not emphasized in the detection output. For this paper, we experimented on five smooth filters: meanshift, conventional bilateral [9], real-time O(1) bilateral [10], median and Gaussian filters. Meanshift and bilateral filters are edge-preserving smooth filters that offer "posterized" or "cartoonized" effects over an input image. Median and Gaussian filters are well-known smooth filters usually used to reduce noises in an image.

Assuming that parameters of each filter are adjusted so that the best detection result is achieved, Fig. 5 shows the gray scale-maps obtained by applying the five filters over the same histogram-equalized image in the multiscale

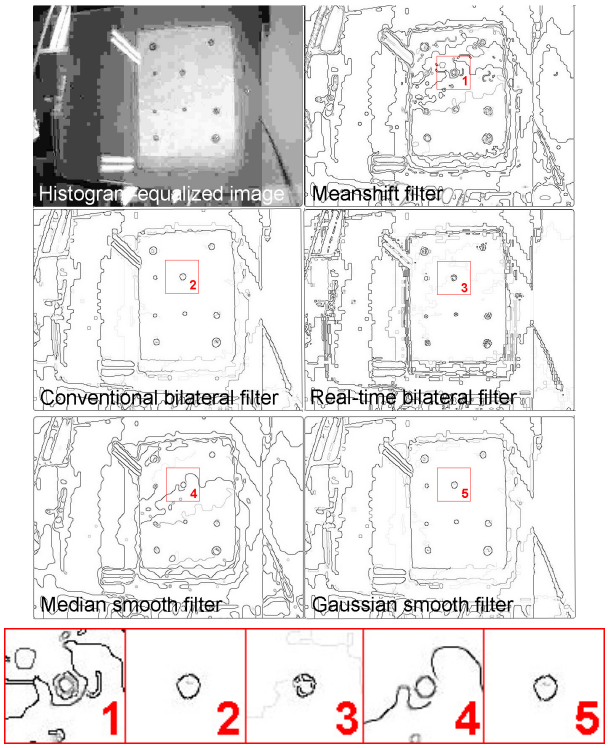

Figure 5. The gray scale-maps of the multiscale Canny edge detection (N = 4) using different smoothing filters. Top-left is the preprocessed image before the multiscale detection is applied.

manner; the number of scales (N) is 4 and the parameters of Canny edge detection are set to $\lambda_1 = 127$, $\lambda_2 = 1000$ and $\sigma = 5$ in all experiments. From Fig. 5, it is obvious that the meanshift and median smooth filters emphasize the textural noises in the scale maps; the conventional bilateral and Gaussian smooth filters offer the cleanest and nicest edges. Compared with the conventional bilateral filter [9], the real-time O(1) bilateral filter [10] can filter out equally the textural noises, but does better in preserving edges and enhancing image contrast (the result regarding this issue is not shown in this paper). However, its results are distracted by noise pixels inside the object so that continuous contours of the objects are barely achieved. Note that only the area inside the projector frustum is considered here.

Furthermore, we examined the computational time used by each filter. Under the same constraints, the computation time used per one smoothing execution (without iteration) is 30, 63, 65, 18, and 15 ms for meanshift, conventional bilateral [9], real-time O(1) bilateral [10], median, and Gaussian filters, respectively. Because our input image here requires strong smoothing effects to reduce the textural noises caused by the histogram equalization, this costs much time in the smoothing processes. When dealing with other scenarios of less severe light conditions, reducing the strength of the smoothing filter is allowed and less computation is required. However, considering the time as well as the resulting scale map, the Gaussian smooth filter

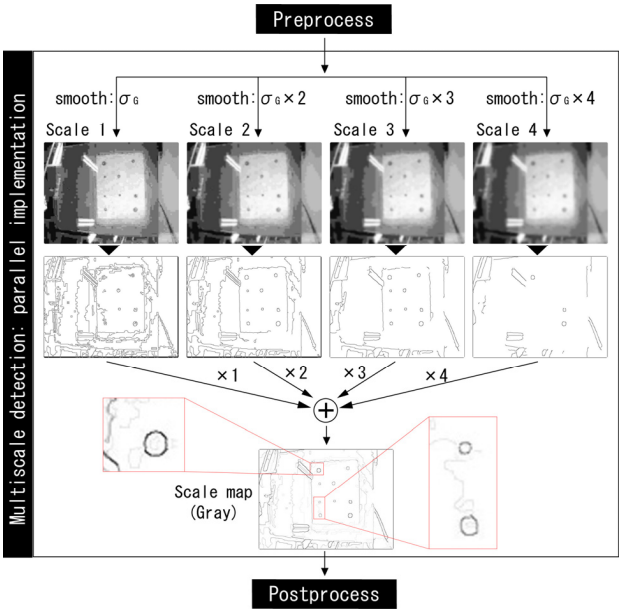

Figure 6. Overview of the investigated multiscale visual object detection using a parallel implementation. The Gaussian smooth filter is used in the multiscale detection.

seems to be the best choice for us. It is not too sensitive to textural noises, provides nice contours of the true objects, and requires reasonable computational times.

C. *Postprocess*

Up to this point, we have achieved a gray scale-map of the multiscale detection. The postprocess step is a simple technique of converting the scale map to a binary image and fixing some special cases. Converting a grayscale image to a binary image is done by using a fixed threshold value so that low-magnitude pixels, which usually belong to noise, are eliminated from the resulting edge map.

Our experiments revealed that the proposed multiscale detection sometimes fails to extract true edges because the magnitude of true edges in the scale map is not high enough. This refers to the cases when the original input image is sufficiently illuminated (e.g., the bottom row of Fig. 2). In these cases, equalizing the image histogram degrades the image contrast and results in failure of the multiscale detection. Setting a lower threshold value during binary conversion can solve this problem but will allow more noises in other images of lower luminance.

We solved this problem by applying the patch image to the binary scale-map, and fulfilling missing edges. The patch image is a binary edge image obtained by applying the base detector with constant parameters (i.e., Canny edge detector with $\lambda_1 = 127$, $\lambda_2 = 500$, and $\sigma = 5$) directly to the input image without any preprocessing. Because the input images focused in these cases are well illuminated, the direct detection result is always clean and accurate (as shown in the third column of the bottom row in Fig. 2). As a result, it may be said that true edges are detected in any arbitrary light condition. Note that this patch image does not affect other insufficiently illuminated input images because parameters used in this patch detection are not sensitive enough to obtain edges in the low-contrast input, as shown in the (edge) patch image of Fig. 3. Besides, the patch image may not be required if using base detectors other than the Canny edge detector.

D. *Parallel implementation*

The multiscale detection described in Section III.B involves a conventional implementation that sequentially passes the simplified image from the higher scale to be the input image in the lower scale. Therefore, the total number of scales is the key factor to determine the computational time of the detection. From our experiments, four scales are adequate; however, the processing time of 137 ms per one frame (based on the Canny edge detector and the Gaussian smooth filter) is far from being a real-time interactive system.

Considering recent growth in multicore processors and parallel programming languages, we changed the sequential implementation to the equivalent parallel implementation. The concept of our parallel implementation to the multiscale visual detection is illustrated in Fig. 6. The preprocessed image is distributed simultaneously to all scales. Instead of using the same smoothing parameters as the sequential implementation, the parallel implementation increases the smoothing effect by enlarging directly the value of σ_G used in each scale. Our experiments presented in Fig. 6 show that the parallel implementation offers similar multiscale detection outcomes compared with the sequential implementation. In this way, speed of detection can be improved significantly with few modifications.

IV. Experimental Results

In this section, we discuss experiments conducted to evaluate the proposed detection scheme. All experiments were performed using an HP Pavilion dv5 Notebook PC with an Intel® Core™2 Duo CPU P8600 running at 2.40 GHz. All images were captured from the beam splitter-based design described in [8] and the projector's focus was adjusted manually in all experiments.

First, we investigated performance of the proposed scheme using the parallel implementation in differently illuminated images. Fig. 7 shows the experimental results. All experiments share the same settings (i.e., $N = 4$ and $\sigma_G = 7$), and no further parameter adjusting is performed in each experiment. According to Fig. 7, it can be seen that the proposed scheme is capable of detecting true edges inside the projector frustum area in all images. Noise edges are visible inside the detected objects given the reflection of the projector's light upon the object's surface. As mentioned earlier, the proposed approach can be applied to other pro-cam systems, regardless of the design in [8]. Nevertheless, some parameters may need to be reconsidered offline to ensure that the smoothing strength and the detector's

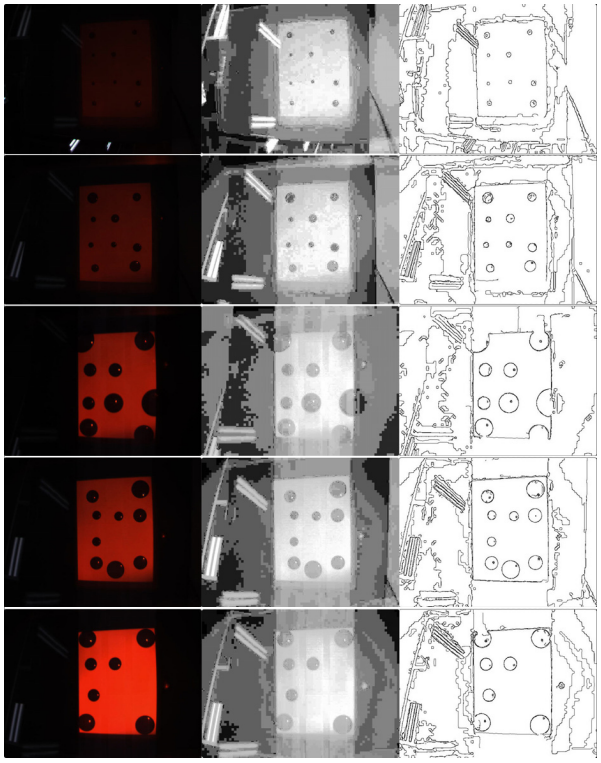

Figure 7. Images in the left column were captured by the beam splitter based pro-cam design proposed by [8]. Images in the middle column are the preprocessed images. Images in the right column result from Canny edge detection using the proposed multiscale approach with parallel implementation.

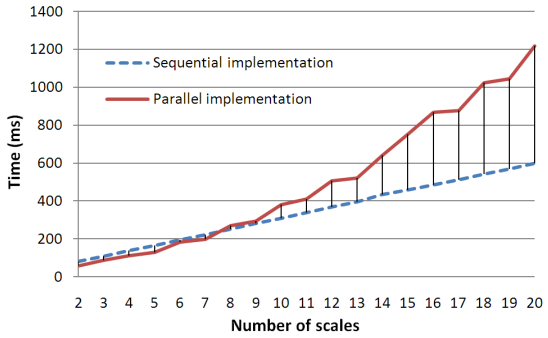

Figure 8. Relationship between the computational time (per one input image) and the number of scales (N) using the sequential and parallel implementation of the proposed detection scheme.

sensitivity are set appropriately.

To determine the speed efficiency of the parallel implementation, Fig. 8 shows the computational time used during the multiscale detection in the sequential and parallel implementations versus the number of scales (N). Our parallel implementation used the OpenMP parallel programming architecture. Therefore, the parallel capability is limited to our Core™2 Duo CPU. According to Fig. 8, the times used by the sequential implementation increase linearly while the times seem to increase exponentially in the parallel implementation. The parallel implementation uses less time than the sequential does when the number of scales is less than 8. This can be explained by the concept of parallel programming in which the ratio of the number of data transfers to the number of executed commands must be small; otherwise, executing in a sequential manner on a single CPU is faster. Higher numbers of scales offer more stable detection results; however, finding appropriate values of smoothing and thresholding may be troublesome. From our experiments, the number of scales from four to six is adequate and provides good balance between detection accuracy and computational time. Hence, using OpenMP programming with a recent processor providing more than two cores should allow for easy acceleration and show clearer differences between times used by the two implementations. Another alternative is to use GPU programming, but major reimplementation of the existing program may be required.

V. Conclusion

In this paper, we explained problems of using visual detection in a portable pro-cam system and investigated a multiscale visual detection scheme as a solution for robust object augmentation applications (especially those operating in a dark environment). The proposed scheme can be used for any vision-based pro-cam application to ensure that good detection results are achieved automatically without the user's guidance regarding light variation in the environment. The investigation mainly involves a sequential multiscale implementation with five different smoothing filters, histogram equalization, and alternative parallel algorithm implementation. The representative based detector used in this paper is the Canny edge detector, but it can be replaced with other visual detectors with behaviors that match the requirements of a specific pro-cam object augmentation application.

References

[1] X. Cao, and R. Balakrishnan, "Interacting with dynamically defined information spaces using a handheld projector and a pen," Proc. UIST, 2006, pp. 225–234.

[2] K. Hosoi, V. N. Dao, and M. Sugimoto, "CoGAME: Manipulating by projection," Proc. ACM SIGGRAPH Emerging technologies, 2007.

[3] M. Kanbara, A. Nagamatsu, and N. Yokoya, "Augmented reality guide system using mobile projectors in large indoor environment," Proc. UbiProjection, 2010.

[4] P. Mistry, and P. Maes, "SixthSense: a wearable gestural interface," Proc. ACM SIGGRAPH ASIA, 2009.

[5] D. Molyneaux, H. Gellersen, G. Kortuem, and B. Schiele, "Cooperative augmentation of smart objects with projector-camera systems," Proc. UbiComp, 2007, pp. 501–518.

[6] Open Source Computer Vision (OpenCV), http://opencv.willowgarage.com/wiki/.

[7] R. Raskar, J. V. Baar, P. Beardsley, T. Willwacher, S. Rao, and C. Forlines, "iLamps: Geometrically aware and self-configuring projectors," ACM Trans. of Graphics, vol. 22, 2003, pp. 809–818.

[8] T. Siriborvornratanakul, and M. Sugimoto, "ipProjector: Designs and techniques for geometry-based interactive applications using a portable projector," IJDMB, vol. 2010, 2010.

[9] C. Tomasi, and R. Manduchi, "Bilateral filtering for gray and color images," Proc. IEEE ICCV, 1998, pp. 839–846.

[10] Q. Yang, K. H. Tan, and N. Ahuja, "Real-time O(1) bilateral filtering," Proc. IEEE CVPR, 2009, pp. 557–564.