A Multi-View Camera System for The Generation of Real-Time Occlusion-Free Scene Video

Alparslan Yildiz and Yusuf Sinan Akgul

{yildiz, akgul}@bilmuh.gyte.edu.tr
GIT Vision Lab
Department Of Computer Engineering
Gebze Institute Of Technology
Cayirova, Gebze, Kocaeli 41400
Turkey

Abstract. This paper presents a novel multi-view camera system that produces real-time single view scene video which sees through the static objects to observe the dynamic objects. The system employs a training phase to recover the correspondences and occlusions between the views to determine the image positions where seeing through would be necessary. During the runtime phase, each dynamic object is detected and automatically registered between the views. The registered objects are learned using an appearance based method and they are later used to superimpose the occluded dynamic objects on the desired view. The occlusion detection is done using a very efficient and effective method. The system is very practical and can be used in real life applications including video surveillance, communication, activity analysis, and entertainment. We validated the system by running various tests in office and outdoor environments.

1 Introduction

The video produced by a camera with a 2D sensor array, i.e., the 2D video of a scene, is not sufficient to represent all the dynamic information about the 3D world. Yet, using 2D videos of real scenes to obtain dynamic real world information is very common in Computer Vision research and everyday life because of the wide availability and low cost of 2D cameras. One partial solution to the insufficiency of 2D videos is to use a multi-camera system to gather more than one 2D video of the scene from different angles. However, this solution introduces new problems such as occlusion analysis and it also makes the system more complex in terms of automatic or manual processing. To handle this complexity, it is desirable to produce a single "occlusion-free" 2D video of the scene using the images from a multi camera system. In order to establish a see through effect, the occluded dynamic objects of the scene would be clearly superimposed on the static scene objects that cause the occlusions. Such a system would be very useful both for automated and manual processing. For example, there are

monocular object tracking systems for which occlusion is a serious problem. The blob based method of Haritaoglu *et al.* [3] uses a monocular method for the surveillance of people. Since it uses only 2D images from a single view, it relies on the detection of the blob merging for the occlusion detection. These types of tracking systems would benefit from our monocular "occlusion-free" scene images where the occluded humans are already marked clearly. Similar monocular object tracking methods include techniques that use object contours for occlusion detection, e.g., [5], and techniques that use depth ordering for the occlusion detection, e.g., [9]. An "occlusion-free" scene image would also be very useful for direct human interaction for entertainment, for communication, and also for manual video surveillance. For example, if we consider a security personnel of a department store, it might cause fatigue or some critical delays to switch between views of the store security cameras continuously. Using a single "occlusion-free" view to eliminate switching between views might be a better alternative for the manual activity tracking.

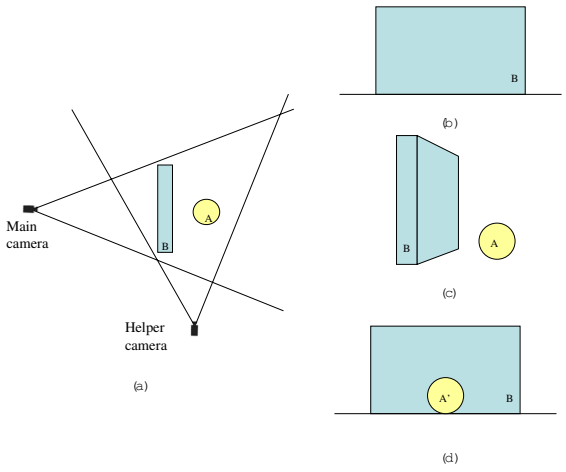

Fig. 1. (a) A scene containing a wall and a dynamic sphere. (b) View of the scene from main camera (c) View of the scene from second camera. (d) "Occlusion-free" view of the scene for the main camera view

In this paper, we present a novel system that uses a multi-camera setup to gather information about a scene and produce an "occlusion-free" view of the scene where the dynamic occluded objects are clearly marked (See Figure 1). The system is very efficient and it can work in real time on a general purpose home computer. The presented system includes an off-line training about the scene that will recover all the occluded regions. After the training, the system can work in real time to produce a 2D "occlusion-free" video of the scene from the point of view of one of the cameras.

There are systems in the literature addressing the occlusion problems with the multi-view camera setups. One category of such systems [8, 4] uses the technique called synthetic aperture focusing, which requires about 100 cameras to simulate a very large physical aperture. Since large apertures receive light rays from many directions, it is possible to eliminate small occlusions and create new "see through" images. Our system is fundamentally different from these systems because we use only a few cameras instead of hundreds. Even though our system requires an off-line training about the scene, it is efficient enough to work in real time after the training. Finally, we do not place any restrictions on the occlusion size.

The rest of this paper is organized as follows: The detailed description of the proposed system is provided in Section 2. We provide the experimental results of the system in Section 3. Finally, we include a discussion about system usage possibilities, system limitations, and concluding remarks in Section 4.

2 "Occlusion-Free" Video Generation

Our approach in producing "occlusion-free" videos is based on using one or more helper cameras (i) to detect if a dynamic scene element is occluded on the main view, and (ii) to use the previously learned appearance of the occluded dynamic object image to superimpose on the main view. This means that the "occlusion-free" video will be produced from the viewing angle of the main camera. There are three basic assumptions of the system about the real world: (i) all the dynamic objects are always in contact with a 3D plane Π such as the ground, (ii) the static objects of the scene are stationary and the occlusions are always caused by the stationary objects, (iii) and the camera image planes have one of their axes roughly parallel to the plane Π. Although these assumptions might be seen as limitations of the system, they hold for a very wide range of real world applications because the scene elements that cause the occlusions are usually stationary, such as the walls of aisles in department stores, vertical columns in buildings, heavy furniture in office environments. In addition, almost all work environments have planar grounds and the cameras can always be positioned with their X axes parallel to the ground.

The proposed system has two phases: training time and run time. The training is responsible for recovering the occluded image regions between the views. It also finds a perspective transform M_{ij} between the plane Π_i of helper camera i and the plane Π_j of the main camera j.

The direct way to perform the training of the scenes is to recover the 3D structure first using calibrated cameras. The recovered 3D structure would produce the occluded image regions and the perspective transforms. However, this approach is very difficult to implement in real life because establishing correspondences between two cameras with very different view points is not a trivial task. There are techniques in the literature that recover the occluded image regions directly such as the work of Zitnick and Kanade [10], but such techniques are

not very helpful in our case because we need to find the image regions between views that correspond to occluded areas.

We developed a structured light solution for a more practical recovery of the occluded regions and their corresponding matches in the other views. Although, the structured light solution works very well indoors, for outdoor applications we need to switch to a laser scanner solution or the occlusion detection needs to be done manually.

2.1 Finding Occluded Areas Using Structured Light

Structured light has been used extensively to establish correspondences between camera pairs or between camera-projector pairs. One class of structured light methods, e.g. [2], uses color coding to project patterns on the scene, and the projected color patterns are used to find correspondences. These types of systems assume that the scene does not change the colors of the projected patterns. The other type of structured light systems project different patterns over time on the same scene under the assumption that the scene stays stationary. Gray codes[1] are one of the most popular among these methods and we employ an adaptation because our training stage is stationary and these methods are more robust against different types of objects in the scene. Our structured light method is similar to Scharstein and Szeliski[7]. The main difference of our work is that we use coded maps to resolve occlusion matches because our camera angles are very different compared to the stereo cameras used in [7]. Figure 2(a-d) shows the result of this process for a scene with two occlusions (showing only low-order unique code bits).

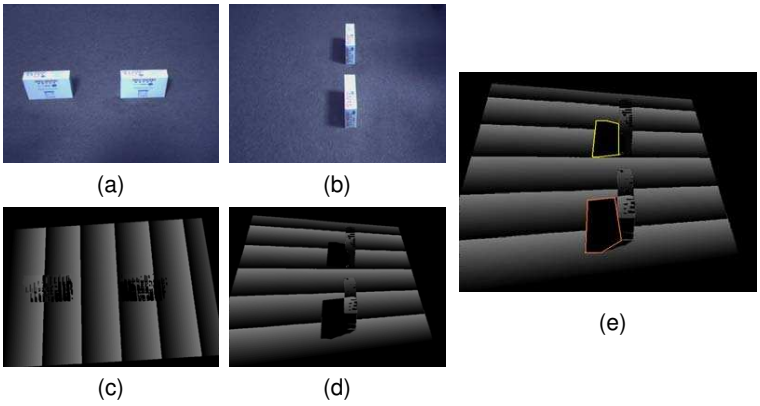

Fig. 2. (a) Main view of the scene with two occlusions. (b) Helper view of the scene. (c) low order u codes for the main view. (d) low order u codes for helper view. (e) Occluded areas found on the helper view.

Finding the occluded areas and their matches in the other images is the major task of the training phase. Although it is not required for a general solution, our occlusion detection method becomes very practical and robust if the structured light source is placed very close to the main view camera position with a roughly similar orientation. By using the unique codes, we find the occluded areas and the shadows which are the image regions where the unique codes cannot be found due to the occlusions. To find the occluded areas on the helper views, we find the *shadow area* O_i^m on unique code-map of each helper view i. We fit convex a hull for each O_i^m as shown in Figure 2-e of a scene with two occluded regions. We also use a minimum threshold value on the area of the regions so that we do not match shadows due to noise and other effects. If more than one projector is used, the structured light sources can be placed at other positions in the scene.

The last step of the training process is to find the perspective transformation between the views so that at runtime we can process the data that comes from the other views.

2.2 Perspective Transform Between Views

We need to define the mathematical relationship between views so that during the runtime if an occlusion of a dynamic object is detected, the information about that object can be brought from the other views. A perspective transform is sufficient to explain this relationship because our system assumes that all objects are in contact with the ground plane as explained in Section 2. Once the perspective transform between the images of the planes on two different views is estimated, the position of the dynamic occluded object on the main view can be estimated using the position of the object in the other views. We can estimate the perspective transform between the views using at least four corresponding points on the images of the plane. Note that it is trivial to automatically pick unique codes that belong to the plane by choosing points near the occluded regions. If there is no structural light available as in outdoor scenes, it is also a very convenient task to pick these points manually because this process will be done only once during the training time.

To find the perspective transform M_{ij} from view i to view j, we use

$$[wx\ wy\ w]_j^T = M_{ij}[x\ y\ 1]_i^T, \tag{1}$$

where $[x/w\ \ y/w]_j$ is a point from main view j and $[x\ \ y]_i$ is the corresponding point from helper view i. One can select more than four correspondences and solve the overdetermined linear system using least-squares to calculate the perspective matrices for a more robust solution.

2.3 Run-Time

During the run time, the system performs a two step algorithm for each frame from all helper views. First step is to find blobs and decide whether blobs are inside any occluded area. Second step is to decide what to do with the blob.

First, the blob B_i^k that belongs to the dynamic object k is found on the helper view i, which can be done by subtracting the estimated background image from the current frame of the view i. Due to the system assumption of the planar environments, the dynamic object k has to intersect the ground plane Π_i at least in one 3D position. The projection of this position on the helper view i, which is called *the ground touching point*(*gtp*), p_i^k, can be found easily: it is the pixel within the object blob B_i^k with the highest y value, assuming the coordinate center is on the upper left corner of the frame. It is guaranteed that $p_i^k \in \Pi_i$ and $p_i^k \in B_i^k$, so if the position of this pixel is transformed using the perspective transformation matrix between the helper view i and the main view, we can obtain the corresponding point on the main view. The corresponding main view point is calculated with the equation

$$p_j^k = M_{ij} p_i^k, \tag{2}$$

where p_i^k is *gtp* of the blob B_i^k on view i, and p_j^k is the corresponding point on view j, which is chosen to be the main view. Note that at this time, for a given pixel $r \in B_i^k$, the system knows if it is occluded by checking if $r \in O_i^m$ for all m as explained in Section 2.1.

As mentioned before, having all the occluded regions O_i^m at hand is not necessary. The perspective transformation will help to check if a blob is occluded or not. Once a blob B_i^k is found on the helper view i, its *gtp* p_i^k can be transformed to the main view to check if there is any blob standing where the transformed *gtp* points on the main view. If there is, then it is decided that the blob B_i^k is not occluded, otherwise it is occluded. This also makes it possible to adapt the system to changing occlusions and to changing backgrounds. To keep the things simple and to gain speed on computation, the occluded areas are found at training phase in our current experiments.

The second step is to decide what to do with the blob B_i^k and will work differently for the occluded blobs and non-occluded blobs. Let us first consider that object k is not standing on the occluded area, i.e., $p_i^k \notin O_i^m$ for any m. This means that the object k is visible from both the view i and the main view j. The system will now memorize both appearances B_i^k and B_j^k of the object k as a pair. Later this memory of appearances will be used to estimate the main view appearance of an occluded blob. To memorize, first, the system finds the corresponding object, if there is any, on the main view. It is sufficient to search the neighborhood of the point p_j^k (Equation 2) on the main view to find the corresponding blob B_j^k. If there is such a blob, then B_i^k and B_j^k are registered as a pair $\left(B_j^k, B_i^k\right)_t$ for the time frame t, resizing all blobs into a fixed bounding box. Note that, if we have more than one helper camera and if the same blob is visible from more than one helper view, then the above pair becomes an ordered set. Note also that, this process is actually a simple appearance based object recognition learning technique.

Let us consider the other case where object k on the helper view i is on the occluded area, i.e., $\exists O_i^m$ and $p_i^k \in O_i^m$. It is obvious that there would be no matching blob on the main view at the corresponding position shown by the

transformation (Equation 2). However, this position is very important because it marks the image region where the object k would have been shown if there were no occlusions. This means that if we had the appearance of the occluded dynamic object k from the main camera angle, we could have superimposed it at this location. Unfortunately, we do not have the occluded object image from the main camera view at this time instant because the object is only visible from the helper camera(s) and it is occluded for the main camera. However, it is very likely that the main camera has seen the dynamic object k many times while it was not occluded, which means that the system has a number of image pairs for this object in the memory. Therefore, making a search on the image pairs in the memory would retrieve this image. An optimization with the formula

$$min\ S(B_i^k, B_i(t,l)), \tag{3}$$

performs this search to produce the image pair whose first component includes the image that we need to superimpose on the main view. $S(B_i, B_j)$ is a function that takes two blob images and returns a similarity value. $B_i(t,l)$ iterates over the second component of the blob pair $\left(B_j^l, B_i^l\right)_t$ for all blob pairs of view i, all frames t and all blobs l. If the result from Formula 3 is lower than a threshold value, then the system does not have the image of the occluded object from the main camera view. In this case, the image to be painted on the main view can be chosen as the image of B_i^k. In other words, the blob detected on the helper view can directly be drawn on the main view at the position where the object would be standing. The direct application of the above method causes blinks in the generated video because the consecutively superimposed images might not be continuous. A considerable improvement for this approach is implemented in our system: it is known that moving objects like humans make periodical and continuous movements. Therefore, when the system does not have the image of the occluded object from the main camera view, the consequent main camera view appearances for the last match could be used for superimposing for most of the cases. In other words, given two image sequences starting with similar frames, we expect that these sequences will include similar images for next few frames. In our experiments, 5 to 10 consequent frames are superimposed from the same sequence if we cannot find any matches. If the system still cannot find a suitable match, the helper view blob B_i^k itself is painted on the main view.

The activity detection and runtime module has to be very efficient because our system needs to process frames in real time. As a result, the blob storage and retrieval functions cannot be very complex. So, we used a basic Sum of Squared Differences (SSD) approach to implement the function S of Formula 3 due to its efficiency, as the registered pairs are at fixed size. Since the system has the blobs(masks) and appearances, only the pixels within the intersection of silhouettes of appearances are taken into SSD computation. Other alternatives for this effective appearance based approach would include employing a full scale object recognition module which would hurt the performance of the system in terms of running times.

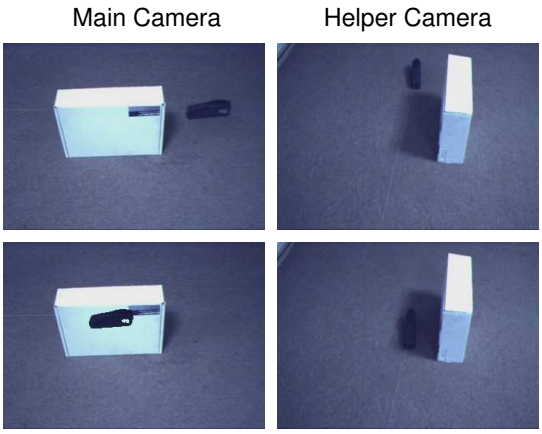

Fig. 3. Indoor experiment: A stapler object is pulled by a string behind an occlusion.

3 Experiments

We tested our system both for experimental office environments and for more challenging outdoor environments. The office setup uses two CCD cameras and an ordinary projector to project patterns onto the scene. For the outdoor setup, however, we entered the plane Π correspondences and the occluded region O_i^m parameters manually because ordinary projectors are not very effective outdoors. Using a field laser scanner would achieve the same task if automatization is needed as in [6]. Figure 3 and Figure 4 shows some selected frames from the output of the experiments.

Visual inspection of the results indicates that the system works very well for the more controlled office environments. All the occluded frames are handled very nicely and the occluded object positions are superimposed at the correct positions. The system also performs favorably for the outdoor experiments but there are some improvements that need to be made. Most of the problems with the outdoor experiments actually come from the blob detection phase which has to be very fast. If the blobs are detected correctly, then the rest of the system works very well. We observed that Formula 3 returns an acceptable appearance to be superimposed on the main camera view for most of the frames. We also observed that the system can process more than 15 frames per second for almost all the experiments on a regular computer.

4 Discussion and Conclusions

We presented a multi-view camera system that produces real-time "occlusion-free" videos of scenes with stationary occlusions. The system is applicable in

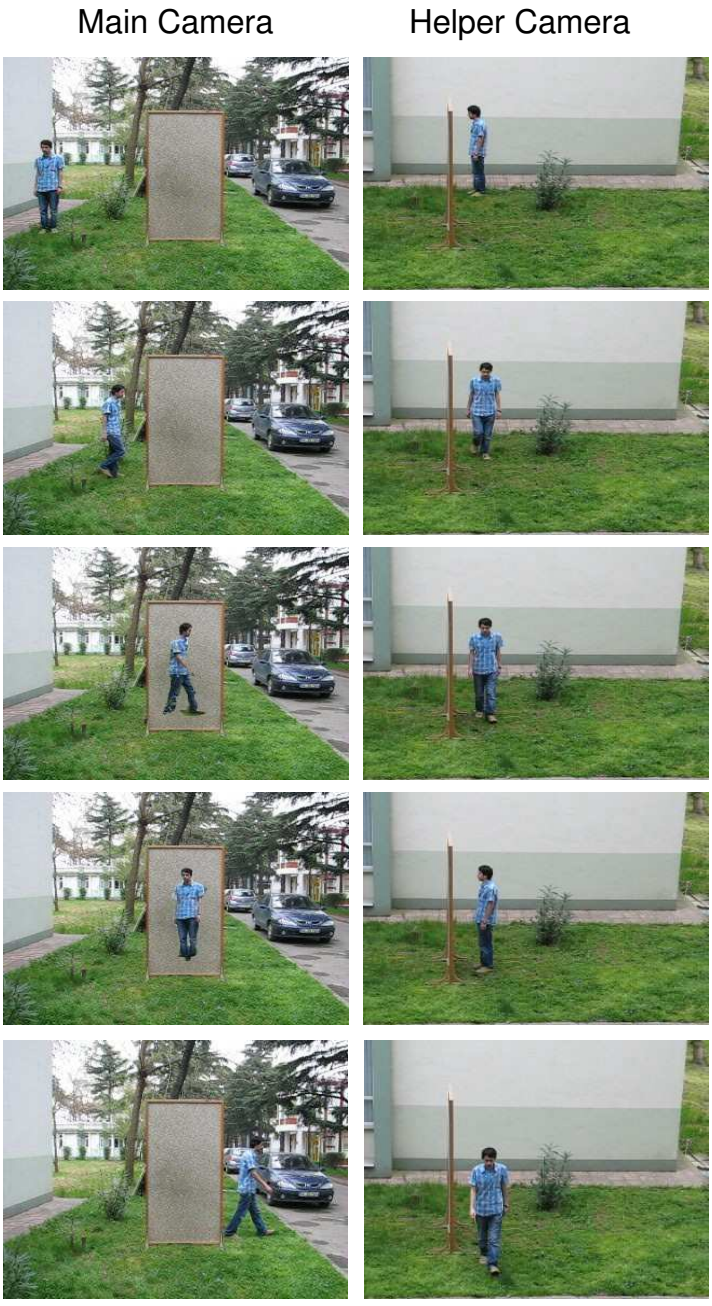

Fig. 4. Outdoor experiment: A person passing behind an occlusion.

many practical tasks, such as in automated and manual video surveillance, entertainment, and human activity analysis. The system has several components each of which is optimized for speed because of the real time requirements.

We are working on several system improvements. The system currently uses a simple activity detection algorithm, which can be improved to include any gradual changes in the scene. It is also possible to handle changes in the static environments by double checking if a detected blob has a corresponding blob on the main view instead of an occlusion. Such an improvement is feasible and it can easily lead to the elimination of the training phase. It is also possible to improve the appearance based blob recognizer module of the system by utilizing a more sophisticated and efficient storage and retrieval system.

Overall, we are very encouraged with the current results and we think that this system will find many real world applications with great success.

References

1. Bitner, J. R., Erlich, G and Reingold, E. M, Efficient Generation of the Binary Reflected Gray Code and its Applications, Communications of the ACM, Vol 19, 9, 1976
2. Davies, C. and Nixon, M., Sensing Surface Discontinuities via Coloured Spot., Proc. IEEE International Workshop on Image and Signal Processing, 1996
3. Ismail Haritaoglu, David Harwood, and Larry S. Davis. W4: Real-time surveillance of people and their activities. IEEE Trans. Pattern Anal. Mach. Intell., 22(8):809-830, 2000.
4. Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. Dynamically reparameterized light fields. In SIGGRAPH, pages 297-306, 2000.
5. John MacCormick and Andrew Blake. A probabilistic exclusion principle for tracking multiple objects. International Journal of Computer Vision, 39(1):57-71, 2000.
6. M Rioux. Digital 3-d imaging: Theory and applications. In Symposium on Photonic and Sensors and Controls for Commercial Applications, Boston, pages 2-15, 1994.
7. Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In IEEE Computer Vision and Pattern Recognition (1), pages 195-202, 2003.
8. Vaibhav Vaish, Marc Levoy, Richard Szeliski, C. Lawrence Zitnick, and Sing Bing Kang. Reconstructing occluded surfaces using synthetic apertures: Stereo, focus and robust measures. In IEEE Computer Vision and Pattern Recognition (2), pages 2331-2338, 2006.
9. Ying Wu, Ting Yu, and Gang Hua. Tracking appearances with occlusions. In IEEE Computer Vision and Pattern Recognition (1), pages 789-795, 2003.
10. C. Lawrence Zitnick and Takeo Kanade. A cooperative algorithm for stereo matching and occlusion detection. IEEE Trans. Pattern Anal. Mach. Intell., 22(7):675-684, 2000.