An Improvement of the Occlusion Detection Performance in Sequential Images Using Optical Flow

Jin Woo Choi¹, Jae Seoung Kim², Taeg Kuen Whangbo³

¹ Culture Technology Institute, Gachon University,
1342 Seongnamdaero, Sujeong-gu, Seongnam-si, Gyeonggi-do, 461-701, Korea
cjw49@paran.com
², ³ Department of Computer Science, Gachon University
rememberguy@hanmail.net, tkwhangbo@gachon.ac.kr

Abstract: During the process of acquiring motion vectors through existing optical flow, problems occur due to occlusion. To solve this, the flow value obtained through optical flow is used to determine the occlusion candidate region. In addition, a bilateral filter, which is a blurring filter that preserves the edge data of an object, is used to repeatedly perform the flow compensation process to more accurately detect occlusions.

Keywords: Occlusion Detection, Optical Flow

1 Introduction

We are designing an algorithm that automatically produced a depth map using motion vector and vanishing point information, but it is limited by the low accuracy of motion vectors due to an occlusion region in the optical flow process that acquired motion vectors. The objective of the present study is to more accurately detect the occlusion region from current and next frames compared to our previous approach, thereby creating accurate motion vector values in the optical flow process through flow value compensation in the occlusion region. We examined the presence of occlusion around the object boundary through object segmentation and optical flow and herein propose a method of increasing the accuracy of occlusion region detection through postprocessing of the occlusion region.

2 Proposed Method

We set the following objectives to solve the problems described above:

• Occlusion is not tested in all regions in current and next frames but occlusion candidates are narrowed into a boundary region of object where most occlusion occurs, thereby improving processing speed and preventing noise generation.
• If the occlusion region is determined based on pixels, it cannot detect occlusion or noise, which causes interruption. Therefore, the occlusion region is objectified through postprocessing, and nondetected occlusion points are filled later.

![Flow chart of the proposed algorithm](image)

**Fig.1.** Flow chart of the proposed algorithm

### 2.1 Watershed algorithm-used object segmentation

Because most occlusion regions occur around boundaries of objects in images rather than occurring inside independent objects, occlusions are not examined in all regions but narrowed to surrounding areas of object boundaries, thereby improving processing speed and preventing noise generation. To achieve this, an object segmentation process is needed, and occlusion candidates are limited to the boundary areas of the segmented object. In this study, we used the rainfall-mode watershed algorithm for object segmentation.

![Method flow Example of marker information used in current and next frames](image)

**Fig.2.** Method flow Example of marker information used in current and next frames

Because our proposed algorithm is characterized by finding occlusion regions using two images of current and next frames, the same segmentation must be done in both frames. To do this, the same marker information and object information that was segmented in the previous frame is used in the next frame as marker information.
A marker uses information of an object boundary segmented in the previous frame. Note that since objects can be moved in images between frames, boundary information is not used directly as a marker but is scaled to a certain ratio based on the object center point. Fig.2 shows a schematic diagram of how to use marker information in current and next frames.

Since boundaries $b_1, b_2$ and $b_3$ are extracted per object as a result of the watershed in the previous frame, marker information $m_1, m_2$ and $m_3$ can be acquired that is scaled down by a certain ratio $k$ based on the center points $c_1, c_2$ and $c_3$ and their respective boundary information. The acquired marker information $m_1, m_2$ and $m_3$ is then used as a marker in the next frame, and nearly the same boundary information $b_1, b_2$ and $b_3$ as the previous frame is acquired.

2.2 Occlusion candidate region determination

In Section 2.1, the object segmentation process was performed using the watershed algorithm in consideration of current and next frames, and objects in videos were separated. Since objects that include occlusion regions have no consistent orientation, the orientation of each object is examined, and objects with inconsistent orientation are classified as “objects that have an occlusion region.” If the objects that have an occlusion region are adjacent to each other, the boundaries of adjacent areas are determined as occlusion candidate regions.

We computed the consistency of motion vectors between current and next frames of pixels that compose each boundary by using optical flow. Since objects that have occlusion regions experience pixel information loss between current and next frames, values of motion vectors acquired through optical flow are not consistent [3][4].

Assuming that an object’s boundary is $E_n$, then angles of vectors obtained through execution of optical flow at points $p_1, p_2, ..., p_n$ of a certain gap can be $\theta_1, \theta_2, ..., \theta_n$. Assuming that variates of vector angles $\theta_1, \theta_2, ..., \theta_n$ are $d_1, d_2, ..., d_n$, if the mean value of the variates is larger than the threshold, an occlusion region in that boundary occurs. This can be expressed as shown in Equation (1). The threshold value is determined adaptively as per the video.

$$\text{if } \frac{\sum_{i=1}^{n} \theta_i}{n} > \text{threshold, occlusion candidate}$$

2.3 Generation of the occlusion point map

If occlusion region candidate boundaries are determined, each surrounding pixel around the candidate’s boundary should be examined to determine if occlusion is present or not. An image of the final examined results of occlusion per pixel is called an occlusion point map. To create an occlusion point map, the difference in intensity value per pixel around the occlusion region candidate boundaries is compared between current and next frames and each pixel's result is filled in the occlusion point map according to whether the occlusion is present or not.
To compare intensity values per pixel, we used the Pyramid Lucas-Kanade Optical Flow method with 5×5 pixels around the occlusion region candidate boundary. The error rate was reduced by referring information from surrounding pixels using a 3×3 block size for each pixel [1]. The intensity comparison algorithm of pixels is shown in Equation (5) [5] as

$$\rho(x, y) = \begin{cases} \frac{(l_1(x, y) - l_2(x + v_x(x, y), y + v_y(x, y)))^2}{\text{threshold}}, & \text{occlusion point} \\ \text{otherwise}, & \text{not occlusion point} \end{cases}$$

In the above equation, $v_x(x, y)$ and $v_y(x, y)$ refer to $x$ and $y$ components, respectively, of motion vector calculated through optical flow at a pixel position $(x, y)$ in the current frame. If the difference in pixel intensity value between the same pixels in current and next frames is larger than the threshold, the corresponding pixel is regarded as an occlusion point [2].

3 Experimental Results and Analysis

In the object segmentation process, which was the first process of the proposed algorithm, the watershed algorithm was employed and information from the previous frame was used as a marker in the next frame. This provided an accurate result because of the requirement that current and next frames should have the same segmentation.

In addition, occlusion regions were not examined in all areas of an image but were limited to boundaries of specific objects, thereby reducing computation. To narrow the occlusion candidate region even more, the abutted boundaries of white objects that were adjacent to the same occlusion region were selected as final candidates where occlusion regions were present.

Fig.3. (a) Resulting image of determination of whether objects have occlusion regions or not (b) Final occlusion boundary (c) Occlusion point map

Since final occlusion boundaries are acquired, pixels in a certain distance from the occlusion boundaries are examined as to whether occlusion is present or not, thereby completing an occlusion point map that represents occlusion per pixel. Fig.3 shows an example of this result.
4 Conclusion

In this paper, the following solutions were proposed to solve the problems in existing studies:

- By limiting the search of the occlusion region to specific boundaries where the occlusion region is likely to be present rather than identifying it from all regions in an image, the processing speed is reduced, thereby facilitating the application to a real-time auto-converting process.
- By reducing the error in which the occlusion region is detected even in areas other than object boundaries where the occlusion region occurs, the accuracy of occlusion region detection is increased.
- Through postprocessing, an occlusion region that cannot be detected in the occlusion point map is detected as a form of a plane, thereby increasing accuracy.

In future studies, occlusion regions that are generated at the outer lines of an object as well as inside an object and occlusion regions at the boundary areas of an image should be detected. Furthermore, if occlusions in images consisting of a curve shape or a combination of straight and curve lines can be detected with objectification through labeling rather than objectification of a clustered occlusion point map through Hough transform, this would increase the performance of occlusion region detection in various image shapes.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2057851)

References