Tracking Multiple Objects Using the Fast Level Set Method

辻 徳生(九州大), Collin Jasnoch(University of Wisconsin), 倉爪 亮(九州大), 長谷川 勉(九州大)
Tokuo Tsuji †, Collin Jasnoch ‡, Ryo Kurazume †, Tsutomu Hasegawa †
† Kyushu University, Hakozaki 6-10-1, Fukuoka
‡ University of Wisonsin, Madison,1415 Engineering Drive Madison, WI 53706-1691

Key Words: *Level Set Method, Adaptive Narrow Band, Contour, Merging, Splitting, Distinction, Distinguishing*

Abstract

The development of detecting moving objects has its basis in the use of active contours [1] also known as snakes. However, such a method can not handle topological changes and lead to the introduction of the Level Set Method [4]. Through the use of the extension velocity field and an adaptive fast narrow band method the LSM method's calculation process time has been reduced [7]giving rise to the Fast Level Set Method. Using the FLSM it is possible to track moving objects using the inertia of the extension velocity field. With the proper use of an inertial based extension velocity field matching of objects can be done and thus an accurate labeling method can be created. This paper first discusses the concepts of tracking moving objects and then discusses a previous research project on the distinguishing of the objects. A labeling method is then proposed with the use of an inertial based extension velocity field. A simulation is run for the situation of separately moving objects.

1. INTRODUCTION

1-1 The Level Set Method

The LSM creates an implicit function with one dimension more than the space the contour is defined in. This function is then updated using a partial differential equation (PDE), which is solved using a numerical method such as an upwind scheme, ADI (Alternating Direction Implicit) method, or the AOS (Additive Operator Splitting) [2], [3] approach.

The contours of interest are defined to exist in the zero level set of the added dimension. The dimension of interest can be described as a matrix of values, such as distance to the nearest zero level set. These values are then integrated numerically using an appropriately assigned speed function on each cell of the matrix. The speed function is designed to force the cells' values to be approximately zero when a contour has reached that cell of the matrix, positive if the cell exists outside of a contour and negative if it exists inside a contour.

This process will however gradually accumulate an integral error, and therefore the data must be reinitialized to calculate the proper values for each cell. Since the speed function and the reinitializing of the data are both time consuming, solutions like the Narrow Band Method (NBM) [4], [5] have been proposed. However, the NBM alone is not efficient enough to handle the needed processing time to assign appropriate labels at a desirable processing speed. Therefore, the proposed labeling method uses an improved version called the Fast Narrow Band Method (FNBM) [6].

1-2 The Fast Level Set Method

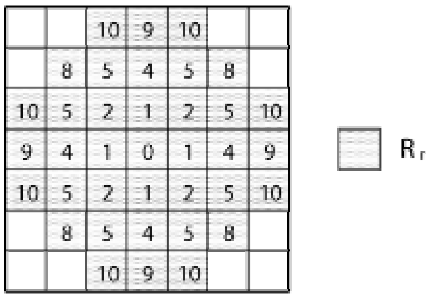
1-2a FLSM Updating Process. The Narrow Band Method [4], [5], uses the fact that updating the implicit function is not necessary to do at great distances from the zero level set. Thus a narrow band can be created around the zero level set to do the update process in. However, the FLSM instead uses an appropriately created algorithm to allow the narrow band to be updated with a concurrent construction of the extension velocity field.

The algorithm used creates reference map comprised of a set of classes R_r, where

$$r = 0 \sim \delta(\delta+1)$$

and δ is the width of the narrow band. Therefore each cell will receive its data according to its class type. For example, the class R_r consists of the cells which are located a distance of $\sqrt{r}$ cells away from the center cell. An example of a reference map $(R_0 \sim R_{12})$ in the case of $\delta = 3$ is shown in Fig.1. With this reference map the extension velocity field can be constructed using the assumption that the speed function at the zero level set has already been determined.

At first, using the class $R_{\delta(\delta+1)}$ in the reference map, all cells which are located $\sqrt{\delta(\delta+1)}$ away from the zero level sets are selected, and the speed function at the zero level sets are registered to these cells tentatively. In the case where a speed function is already registered to a cell, the old speed function is overwritten by the new one. This process is repeated until all the classes in the reference map have been investigated for all zero level sets.



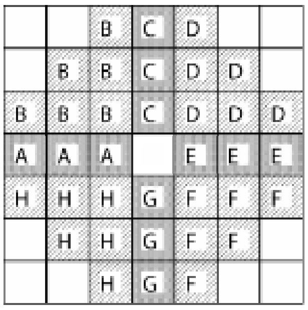
		10	9	10		
	8	5	4	5	8	
10	5	2	1	2	5	10
9	4	1	0	1	4	9
10	5	2	1	2	5	10
	8	5	4	5	8	
		10	9	10		

Figure 1: A reference map

1-2b FLSM Adaptive Updating. It is desirable to only overwrite the appropriate speed functions rather than also overwriting unnecessary cells' functions that will be overwritten again during the same update process. For example, consider a zero level set cell, C_1. If the left adjacent cell, C_2, is also a zero level set cell, then it can be concluded that there are no cells in which a distance to the cell C_1 is smaller than the distance to the cell C_2 in the left region of the reference map. With this conclusion it is realized that the registration for the zero level set cell, C_1, can be omitted for this region to avoid unnecessary overwriting.

This elimination of overwriting can be implemented in all directions, by dividing the reference map is into eight regions. IN Fig. 2 they are labeled A to F. In this example, the regions A, C, E, and G are defined to be the straight neighborhood regions, and B, D, F, and H are defined as the diagonal neighborhood regions. With this categorization, the extension velocity field is constructed using the following method.

1. If an adjacent cell in a straight neighborhood region (A, C, E, or G) of a zero level set cell is also classified as a zero level set cell, then that cell and all cells along the same straight neighborhood region are classified as ***unnecessary***. In addition, the diagonal neighborhood regions adjacent to the cell in the straight neighborhood region are also classified as ***unnecessary***. For example, if the left adjacent cell is a zero level set cell, then the regions A, B, and H are classified as ***unnecessary***.
2. If an adjacent cell in the diagonal neighborhood region is also a zero level set cell, then that diagonal neighborhood region is classified as ***unnecessary***. For example, if the upper right adjacent cell is also a zero level set cell, then the region D is marked as ***unnecessary***.
3. For the remaining regions which aren't classified as ***unnecessary***, the extension velocity function is registered.



		B	C	D		
	B	B	C	D	D	
B	B	B	C	D	D	D
A	A	A		E	E	E
H	H	H	G	F	F	F
	H	H	G	F	F	
		H	G	F		

Figure 2: Adaptive Reference Map

1-2c Reinitializing of the FLSM. In order to update the implicit function with stability, reinitializing the data (calculating the distance for each cell to the nearest zero level set cell) must be done frequently. However, reinitializing the data is time consuming and slows the over all process of the LSM. For this reason the FLSM integrates the reinitializing of the data into the construction process of the extension velocity field with little additional calculation cost [7].

The construction process of the extension velocity field has the speed function registered in an order of distance from the zero level set. At the same time it is then able to overwrite the distance stored in the reference map creating the distance field with little calculation cost.

2. DISTINCTION OF OBJECTS

2-1 Method Basis

During the imaging and object detection process this method uses the contours of the object to calculate the total area. This area is found to determine if that area of the zero level set should be classified as an object of interest or if it is noise of the function. Upon determining if the location of the zero level set is an object the labeling process begins.

2-2 Possible Situations

2-2a Simple Situations. To achieve robustness and allow for further adaptability of a distinguishing system potential situations have been clarified for the method to look for. This allows the method to consider simple situations unless specific changes to the imaging area have been made such as the number of processed objects. In Fig. 3 the simple situation of two objects moving separately is shown. However, at some point in time one of the objects entered while the other was already being processed as seen in Fig.4. In addition eventually one of the objects is likely to exit while the other remains to be processed such as the situation in Fig. 5.

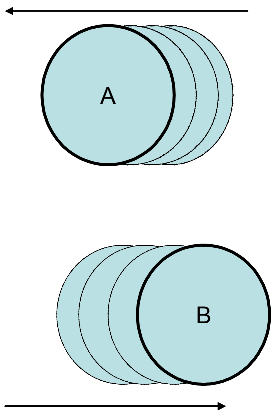


Figure 3: Separate moving objects

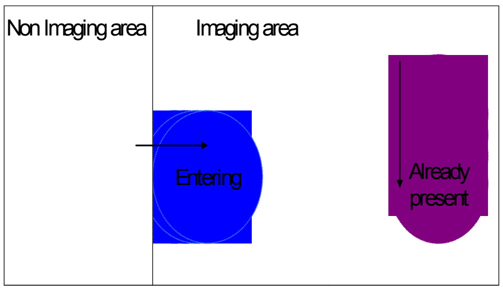


Figure 4: Object entering while object present

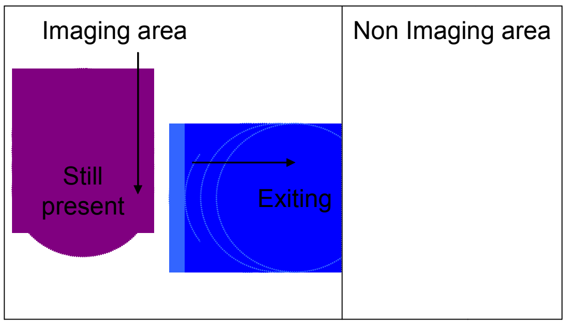


Figure 5: Object exiting while object present

2-2b Merging and Splitting. Comparatively to the situations shown in Figures 4 and 5 are situations of merging and splitting. In the situation shown in Fig. 5 the number of processed objects is being reduced by one. Similarly in a merge situation, such as the one shown in Fig. 6, the amount of objects processed will be reduced by one. Contrast to the above mentioned similarities, the situation in Fig. 4 is similar to the splitting situation such as the one shown in Fig. 7 since the number of processed objects is increasing by one. In both situations of increasing and decreasing of processed objects an approach to determine the differences between their counterpart is required.

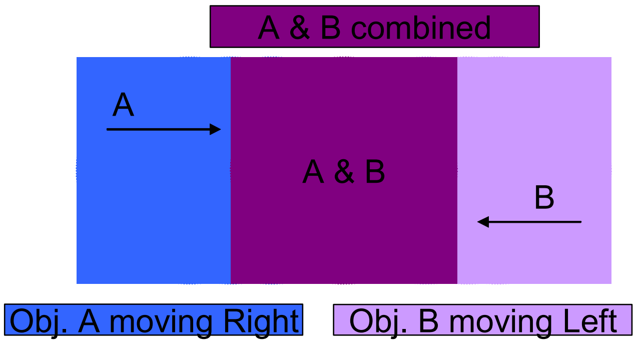


Figure 5: Merging of Objects

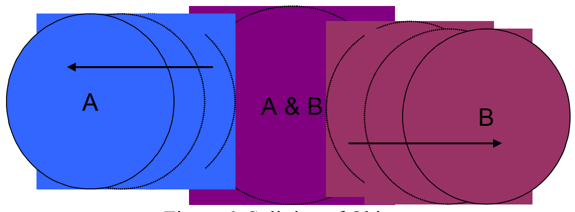


Figure 6: Splitting of Objects

2-3 Central Location Method

2-3a Contour Moments. The previous work for distinction of objects was focused on the contour moments and the realization of the center of gravity for each object. With the knowledge of each objects location a comparison could be made in the next update to find the closest labeled object and update it accordingly.

To handle the situation of merging and splitting the previous method would consider a merge if the object's centers of gravity became to close to each other. Then, while the objects were considered as merged objects a counter incremented to give an accurate number of update cycles the objects had merged. To give appropriate labeling after splitting the previously stored speed of each object (calculated from the centers of gravity) and the number of update cycles absent determined the most likely positions of the objects for matching. However, due to the use of thresholds for the center of gravity in determination of merges and splitting the method occasionally preemptively labels merges or splits.

2-3b Previous Method In Action. In Fig.7a and Fig. 7b there exists an object in the database (labeled 'a'). In these figures a figure can be seen entering on the right side of the imaging area but the FLSM has yet to process it.

In Fig. 7c the FLSM has processed the entering object as well as the object which already existed in the imaging area. The method then successfully retains the appropriate label on the object which was already present (object 'a') and also assigns a unique label to the entering object (object 'b').

In Fig. 7d the labeling method successfully distinguishes the objects and is able to appropriately label them. In addition the method updates any necessary data on the objects being detected.

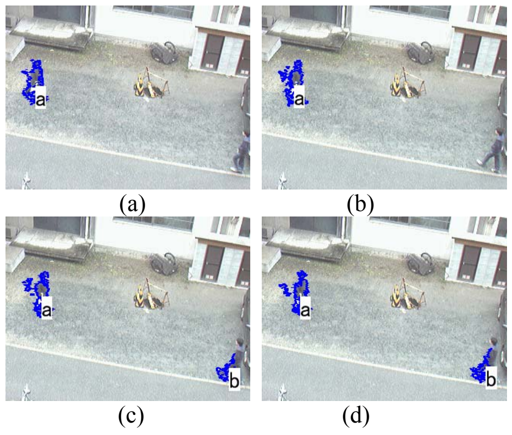


Figure 7: Object entering case B
(a)(b) Updating of object `a`
(c)Entering of object 'b' and updating of 'a'
(d) Updating of both 'a' and 'b'

3 INERTIAL USE OF THE EXTENTION VELOCITY FIELD

3-1 Overview Of Method

The use of locations for the distinction of objects is successful to a certain extent. Instead, a more effective approach is to make use of the speed function used in the calculation of the implicit function. For each object the labeling method stores the local speed function and implicit function to allow for self calculating. Then upon actual updating of the implicit function an accurate match can be made and updated to each object accordingly.

3-2 Self Calculating

3-2a Acquisition Of The Functions. Only certain parts of the speed function and implicit function are necessary for self calculations. If the whole function set were acquired for each object not only would the calculation time be greatly increased but each object would in effect become all objects and thus make it impossible to make distinctions among them.

To acquire the desired data from each necessary function appropriate masks are created in the area of the objects contour. The size of the mask created depends on the data acquisition it will be used for. For example, the implicit function must store data within the contour and at least out to the edge of the narrow band where as the storing of the actual status of the front and narrow band of the object need only be around the contour with a thickness of,

$$NB * 2 + 1,$$

where NB is the width of the narrowband around the front.

3-2b Calculation Process. For each object the self calculating is done using the FNBM (6) to reduce the calculation time. The self calculating begins with the resetting of the implicit function. If the values of the implicit function are grater than zero they are set to 1 and they are set to -1 otherwise. This allows for an easy calculation of the appropriate gradients for actual rewriting of the function. The next process is the re-labeling of the narrow band and the writing of the speed function to the object's matrices. After the writing of the speed function, calculation of the implicit function, and the re-labeling of the narrowband, the process then redraws the object's masks using the self calculated implicit function. This process is continually done for each object until the object has left the imaging area.

Rather than using the most recently updated value for the local speed of the in the creation of the extension velocity field (Fig. 7), the frontal nodes continually average their speed. This averaged value is then used for the creation of the extension velocity field rather then the updated speed functions value. By averaging the front velocity, unique situations such as merging and splitting can be handled since the front velocity can then be used to update the implicit function even after the object has become obstructed. Continual self updating will then allow for an accurate match as long as the objects velocity does not dramatically change during its occlusion.

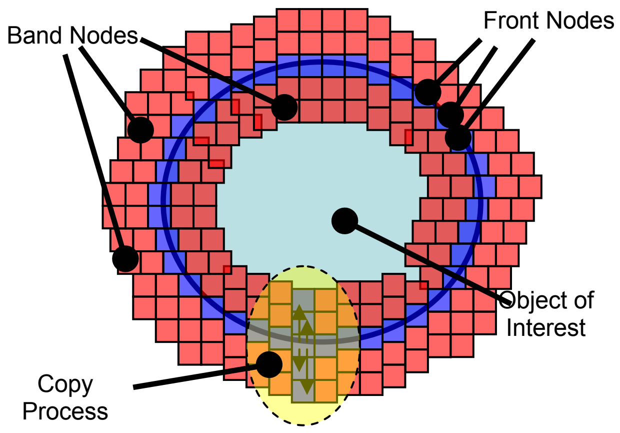


Figure 7: Extension Velocity Field

3-3 Label Map

3-3a Overview. For quick recognition of previously viewed objects versus newly recognized objects a label map is created. This map stores the label of a specific object at the objects front. Using the masks of the actual contours being seen, the label map can be viewed in the desired area to check for a corresponding object.

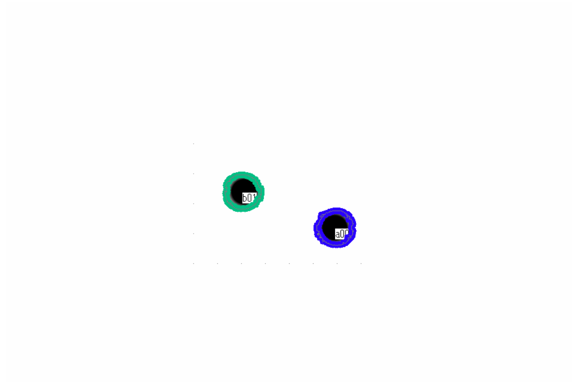
3-3b Creation And Updating. After a contour is determined to be an object, using the mask created for the front of the object (exactly where the contour is), data in the label map can be set to represent the object. Since the data is only copied where the contour is, the actual process time is quite small.
However, the first step in the creation of the label mask is actually the copying of the entire status map. This allows the label map to have distinctions between new objects and previously processed objects. Prior to every labeling run the Label map is reset to contain the actual status map of the narrow band and then each objects label value is added to the map where their self calculated front mask has been created.

3-3c Matching. Since the label map contains both the label values and the status type, new objects can be determined by an observation of a lacking label. Actual matching is a simple process of finding the value stored in the label map where the current contour exists. After the label is retrieved the object can receive an update for its actual location by copying the speed function and implicit function using the newly created masks. Fig 8a through Fig8b show a simulation of the process successfully updating two objects which is shown by using distinctly different colors for the drawing of the contour. The images are exactly one update cycle apart.

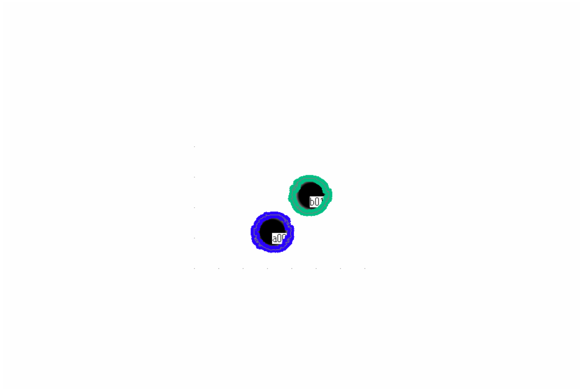
Due to the complex nature of merging and splitting the details of handling such situations have not yet been fully implemented into our program, however such situations will be demonstrated during the allotted presentation time.

4 CONCLUSION

An approach to the distinguishing of multiple merging objects using the inertia of the extension velocity field has been proposed. Through simulation it has been shown to work in situations where the objects are moving separately. The method is proposed to work effectively for complex situations such as merging and splitting as well and can be shown to be effective with further development.



(a)



(b)

Figure 8
(a) Frame 21 of Tracking
(b) Frame 44 of Tracking

References
[1] N. Paragios and R. Deriche. Geodesic active contours and level sets for detection and tracking odd moving objects. *IEEE Tranaction on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000.
[2] T. Lu, P. Neittaanmaki, and X. C. Tai. A parallel splitting up method and its application to navier-strokes equation. *Applied Mathematics Letters*, 4(2):25–29, 1991.
[3] J. Weickert, B. M. ter Haar, Romeny, and M. A. Viergever. Efficent and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, (3):398–410, 1998.
[4] J. Sethian. *Level Set Methods and Fast Marching Methodssecond ed.* Cambridge University Press, UK, 1999.
[5] D. Adalsteinsson and J. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118:269–277, 1995.
[6] A Fast Narrow Band Method and Its Application in Topology-Adaptive 3-D Modeling Shuntaro YUI, kenji HARA, Hongbin ZHA, and Tsutomu HASEGAWA. Proceedings of 16th International Conference on Pattern Recognition (ICPR'02), vol.4 pp.122-125, 2002
[7] Fast Level Set Method and Real Time Tracking of Moving Objects in a Sequence of Images, Ryo Kurazume, Shuntaro Yui, Tokuo Tsuji, Yumi Iwashita, Kenji Hara, Tsutomu Hasegawa; Journal of Information Processing Society of Japan Vol.44, No.8, pp.2244-2254,(2003)