

# Hierarchical framework for direct gradient-based time-to-contact estimation

Berthold K.P. Horn, Yajun Fang, Ichiro Masaki

**Abstract**—The time-to-contact (TTC) estimation is a simple and convenient way to detect approaching objects, potential danger, and to analyze surrounding environment. TTC can be estimated directly from single camera though neither distance nor speed information can be estimated with single cameras. Traditional TTC estimation depends on “interesting feature points” or object boundaries, which is noisy and time consuming. In [13], we propose a direct “gradient-based” method to compute time-to-contact in three special cases that avoid feature points/lines and can take advantages of all related pixels for better computation. In this follow-up paper, we discuss the method to deal with the most general cases and propose a hierarchical fusion framework for direct gradient-based time-to-contact estimation. The new method enhances accuracy, robustness and is computationally efficient, which is important to provide fast response for vehicle applications.

## I. INTRODUCTION

The time-to-contact (TTC) is the time that would elapse before the center of projection (COP) reaches the surface being viewed if the current relative motion between the camera and the surface was to continue without changes. As shown in figure (1), we establish a camera-oriented coordinate system, with the origin at the COP, the  $Z$  axis along the optical axis, and the  $X$  and  $Y$  axes parallel to axes of the image sensor.  $Z$  is actually the distance from the center of projection (COP) to object object. Image coordinates  $x$  and  $y$  are measured from the principal point (foot of the perpendicular dropped from the COP).  $(u, v) = (\dot{x}, \dot{y})$  is the motion field and  $(U, V, W) = (\dot{X}, \dot{Y}, \dot{Z})$  is the velocity of a point on the object relative to the sensor (which is opposite to the motion of the sensor relative to the object) as in figure (1). The velocity  $W = dZ/dt$  is negative if the object is approaching the camera.

The TTC is essentially the ratio of distance to velocity:

$$\text{TTC} = -Z / \frac{dZ}{dt} = -1 / \frac{d}{dt} \log_e(Z) \quad (1)$$

While distance and velocity can not be recovered from images taken with a single camera without additional information, such as the principal distance and the size of the object, the ratio of distance to velocity can be recovered directly, even with an uncalibrated sensor.

B.K.P.Horn, Y. Fang, I. Masaki, are with the Intelligent Transportation Research Center (ITRC), Microsystems Technology Laboratories (MTL), Electrical Engineering and Computer Science Department, Massachusetts Institute of Technology, Cambridge, MA 02139, USA (Email: yajufang@csail.mit.edu, bkph@csail.mit.edu, masaki@mit.edu).

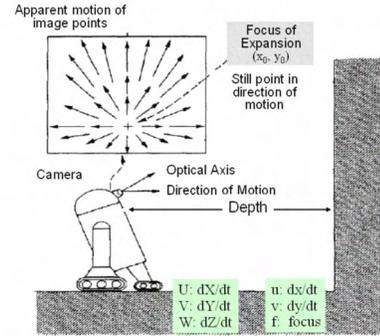


Fig. 1. The definition of Time-To-Contact

### A. The relationship between motion field and time-to-contact/focus-of-expansion

If we consider only the translational model, by differentiating the perspective projection equations with respect to time, we obtain the relationship between motion field and TTC as in equations (2).

$$\begin{aligned} u &= -\frac{W}{Z}(x - x_0) = \frac{(x - x_0)}{\text{TTC}} \\ v &= -\frac{W}{Z}(y - y_0) = \frac{(y - y_0)}{\text{TTC}} \end{aligned} \quad (2)$$

where  $x_0 = f(U/W)$  and  $y_0 = f(V/W)$ , which is the focus-of-expansion (FOE) at which point the motion information is zero. When object point corresponding to FOE moves along the direction of  $(U, V, W)$ , its image projection remains the same. The motion directions at other image points are determined by the vector from the FOE to other image points.

### B. Traditional methods and the limitation

1) *Optical-flow based methods:* A typical approach [1][2][8][9][10][11] to determining the TTC is to take advantage of the relationship between time-to-contact and optical flow as shown in equation (2) and (3).

$$u_x = \frac{1}{\text{TTC}} \quad v_y = \frac{1}{\text{TTC}} \quad (3)$$

Methods for estimating optical flow are iterative, need to work at multiple scales, tend to be computationally expensive and require a significant effort to implement properly.

2) *Size-based methods*: TTC can be estimated by the ratio of the size  $s$  of the image of the object to the rate of change of the size, as shown in equation (4).

$$T = s / \frac{ds}{dt} = 1 / \frac{d}{dt} \log_e(s) \quad (4)$$

High accuracy is needed in measuring the image size of targets in order to obtain accurate estimates of the TTC when it is large compared to the inter-frame interval. When image size is about 100 pixels, and the estimated TTC is 100 frames, then size changes by only 1 pixel from frame to frame. To achieve even 10% error in the TTC one would have to measure target sizes with an accuracy of better than 1/10 of a pixel. Thus the accuracy of TTC computation based on the method is far from satisfying.

### C. Direct method for time-to-contact

In [13], for three special cases (case I)(II)(III) as shown in figure (2), we proposed a direct method [13] based on the “constant brightness assumption” described by equation (11), which needs only the derivatives of image brightness and does not require feature detecting, feature tracking, or estimation of the optical flow.

Define  $C = -W/Z$  (the inverse of the TTC), and  $G = (xE_x + yE_y)$  a short-hand for the “radial gradient” while  $E_x, E_y, E_t$  are the partial derivatives of brightness w.r.t.  $x, y,$  and  $t$ . Let  $p$  and  $q$  be the slopes of the planar surface in the  $X$  and  $Y$  directions.

$$Z = Z_0 + pX + qY \quad (5)$$

Results for the discussed cases are:

- Case (I)

Translational motion along the optical axis towards a planar surface perpendicular to the optical axis;

$$C = - \sum GE_t / \sum G^2 \quad (6)$$

- Case (II)

Translational motion in an *arbitrary* direction relative to a planar surface that is perpendicular to the optical axis;

$$\begin{aligned} & \begin{bmatrix} \sum E_x^2 & \sum E_x E_y & \sum GE_x \\ \sum E_x E_y & \sum E_y^2 & \sum GE_y \\ \sum GE_x & \sum GE_y & \sum G^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} \\ &= - \begin{bmatrix} \sum E_x E_t \\ \sum E_y E_t \\ \sum GE_t \end{bmatrix} \end{aligned} \quad (7)$$

where  $A = f(U/Z)$ ,  $B = f(V/Z)$ , and FOE is given:

$$x_0 = -A/C \quad \text{and} \quad y_0 = -B/C \quad (8)$$

- Case (III)

Translational motion along the optical axis relative to a planar surface of *arbitrary* orientation;

$$\begin{aligned} & \begin{bmatrix} \sum G^2 x^2 & \sum G^2 xy & \sum G^2 x \\ \sum G^2 xy & \sum G^2 y^2 & \sum G^2 y \\ \sum G^2 x & \sum G^2 y & \sum G^2 \end{bmatrix} \begin{bmatrix} P \\ Q \\ C \end{bmatrix} \\ &= - \begin{bmatrix} \sum Gx E_t \\ \sum Gy E_t \\ \sum GE_t \end{bmatrix} \end{aligned} \quad (9)$$

where  $P = (p/f)(W/Z_0)$ ,  $Q = (q/f)(W/Z_0)$ . Thus,

$$p = -f \frac{P}{C} \quad \text{and} \quad q = -f \frac{Q}{C} \quad (10)$$

The proposed methods are so-called “gradient-based” since the matrix elements in the above equations are the function of brightness gradient. The simpler cases (I)(II)(III) are the special situation for case (IV) and all have closed form solutions. The more general case (IV) requires non-linear optimization techniques which will be proposed in the this papers as shown in section II.

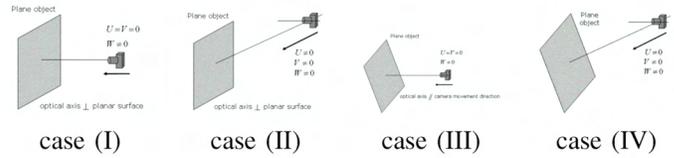


Fig. 2. Four cases of relative motion.

## II. TIME-TO-CONTACT ESTIMATION FOR ARBITRARY TRANSLATIONAL MOTION RELATIVE TO AN ARBITRARY PLANE

This is the most general case as shown in figure (2) where the translational motion needs not be in the direction of the optical axis of the imaging system, and the planar surface needs not be oriented perpendicular to the optical axis.

Substituting the expression for  $Z$  given by equation (5) in expressions for  $u$  and  $v$  given by equation (2) and then inserting these into the brightness change constraint equation (11) leads to

$$uE_x + vE_y + E_t = 0 \quad (11)$$

we have,

$$-\frac{W}{Z_0} \left( 1 - p \frac{x}{f} - q \frac{y}{f} \right) [(x - x_0)E_x + (y - y_0)E_y] + E_t = 0$$

Follow the previous definition for  $A, B, C, P, Q$ , the above equation can be rewritten as:

$$C \left( 1 + x \frac{P}{C} + y \frac{Q}{C} \right) \left[ G + E_x \frac{A}{C} + E_y \frac{B}{C} \right] + E_t = 0 \quad (12)$$

We can formulate a least squares method to find the five unknown parameters  $A, B, C, P,$  and  $Q$  that minimize the following error integral or sum over all pixels of a region of interest (which could be the whole image):

$$\sum \left\{ C \left( 1 + x \frac{P}{C} + y \frac{Q}{C} \right) \left[ G + E_x \frac{A}{C} + E_y \frac{B}{C} \right] + E_t \right\}^2 \quad (13)$$

or

$$\sum [CFD + E_t]^2 \quad (14)$$

where  $F = 1 + xP/C + yQ/C$ ,  $D = G + E_x A/C + E_y B/C$

To find the best fit values of the five unknown parameters we differentiate the above sum with respect to the five parameters and set the results equal to zero. This leads to five equations in five unknowns. The equations are nonlinear and need to be solved numerically.

If  $P/C$  and  $Q/C$  are given, then  $F = 1 + xP/C + yQ/C$  is known and the cost function equation (13) are linear in the remaining unknowns  $A$ ,  $B$ , and  $C$  as followed:

$$\sum [F * (CG + E_x A + E_y B) + E_t]^2 \quad (15)$$

Parameters  $A$ ,  $B$ , and  $C$  can be solved based on the following linear equations:

$$\begin{aligned} & \begin{bmatrix} \sum F^2 E_x^2 & \sum F^2 E_x E_y & \sum F^2 G E_x \\ \sum F^2 E_x E_y & \sum F^2 E_y^2 & \sum F^2 G E_y \\ \sum F^2 G E_x & \sum F^2 G E_y & \sum F^2 C^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} \\ & = - \begin{bmatrix} \sum F^2 E_x E_t \\ \sum F^2 E_y E_t \\ \sum F^2 G E_t \end{bmatrix} \end{aligned} \quad (16)$$

Conversely, if  $A/C$  and  $B/C$  are given, then  $D = G + E_x A/C + E_y B/C$  is known and the cost function equation (13) are linear in the remaining unknowns  $P$ ,  $Q$ , and  $C$ .

$$\sum [(C + xP + yQ) * D + E_t]^2 \quad (17)$$

Parameters  $P$ ,  $Q$ , and  $C$  can be solved based on the following linear equations:

$$\begin{aligned} & \begin{bmatrix} \sum D^2 x^2 & \sum D^2 xy & \sum D^2 x \\ \sum D^2 xy & \sum D^2 y^2 & \sum D^2 y \\ \sum D^2 x & \sum D^2 y & \sum D^2 \end{bmatrix} \begin{bmatrix} P \\ Q \\ C \end{bmatrix} \\ & = - \begin{bmatrix} \sum x D E_t \\ \sum y D E_t \\ \sum D E_t \end{bmatrix} \end{aligned} \quad (18)$$

Thus, given an initial guess  $P/C$  and  $Q/C$ , we can alternately solve for  $A$ ,  $B$ , and  $C$  based on initial estimate of  $P/C$  and  $Q/C$  using equation (16). Then, given the new estimates of  $A/C$  and  $B/C$ , we can update  $P$ ,  $Q$  and  $C$  with equation (18). A few iterations of this pair of steps typically yield a close enough approximation to the exact solution.

As before, the time to collision is the inverse of the parameter  $C$ . If desired, the direction of translational motion, given by  $(U/W) = (A/C)/f$  and  $(V/W) = (B/C)/f$  can also be calculated, as can the orientation of the surface specified by  $f(P/C)$  and  $f(Q/C)$ .

### A. Impact of different assumptions

We have discussed different computational methods based on different assumptions. For the most general situation, there are 3 unknown motion-related parameters,  $(U, V, W) = (\dot{X}, \dot{Y}, \dot{Z})$  is the moving velocity relative to the sensor and 2 unknown object parameters,  $p$  and  $q$ , the slopes of the planar surface in the  $X$  and  $Y$  directions.

For case (I), it is assumed  $U = V = 0$  and  $p = q = 0$ . One unknown parameter:  $W$ .

For case (II), it is assumed  $p = q = 0$ . Three unknown parameters:  $U, V, W$ .

For case (III), it is assumed  $U = V = 0$ . Three unknown parameters:  $W, p, q$ .

For case (IV), no assumption. Five unknown parameters:  $U, V, W, p, q$ .

For case (II), the planar surface is assumed to be perpendicular to the optical axis while the translational motion in an *arbitrary* direction. If the translational motion is also along the optical axis, i.e.,  $U = 0$  and  $V = 0$ , we should have  $A \approx 0, B \approx 0$ , and  $x_0 \approx 0, y_0 \approx 0$  based on equation (8), which shows that the FOE will be around the origin.

For case (III), the translational motion is assumed to be along the optical axis while the planar surface is of *arbitrary* orientation. If the planar surface is also perpendicular to the optical axis, we should have  $p/f = -P/C = 0 \approx 0$  and  $q/f = -Q/C = 0 \approx 0$  according to equation (9)(10).

The physical model of case (I) is a special situation of cases (II), (III) and (IV). Applying the computational method for case (II) for case (I), we should have  $A \approx 0$  and  $B \approx 0$ . Applying the case model (III) for case (I), we should have  $P \approx 0$  and  $Q \approx 0$ . If we use the case model (IV) for case (I), we should have  $A \approx 0, B \approx 0, P \approx 0$  and  $Q \approx 0$ , which shows that the object plane is perpendicular to the optical axis and move along the optical axis. Thus we can identify case (I) in real computation.

### III. SEVERAL FACTORS THAT AFFECT COMPUTATIONAL RESULTS AND HIERARCHICAL TIME-TO-CONTACT ESTIMATION BASED ON FUSION OF MULTI-SCALED SUBSAMPLING

After finding theoretical solutions of TTC estimation, we further discuss several factors that would have impact on the accuracy and reliability of the computational results, which includes: computational models, the threshold on the time derivative, computational areas (full images or segmentation regions), sub-sampling factor, etc.

#### A. Impact of different computational models

Normally the computational model for case (IV) is the best choice when we do not know any prior information about the relative motion. In real application, we apply all four different computational models to estimate TTC values. The computational models for case (I)(II)(III) can provide fast close-form solutions and have other merits while the computational method for case (IV) depends on iteration which involves more computational load and might not converge or might

not provide reliable solutions. The estimation accuracy will be improved if we fuse the results from four computational models.

### B. Impact of computational areas or contributed pixels

The computational methods as introduced in equation (6), (7), (9), (18) need to compute the integral or sum over all pixels of a region of interest. We can sum over either the whole image or specified regions.

TTC computation based on whole images is somehow similar to the weighted average of TTCs for pixels from foregrounds and backgrounds. Typically, background objects are static and far away. The large TTC corresponding to background pixels makes TTC computation based on whole images larger than based on pure foreground pixels. The farther away the foreground is, the larger the background fraction is, the larger the integrated TTC estimation based on full images is.

1) *Impact of threshold for time derivative of brightness:* The impact of large TTC from background pixels can be reduced by setting a threshold on the time derivative of brightness,  $E_t\_Threshold$ , to block contributions of image pixels with minor intensity change and to improve computational accuracy. We tested the impact of different threshold of brightness derivative on TTC estimation between two continuous frames. Results remain relatively consistent unless the  $E_t$  threshold becomes too large. (When  $E_t$  threshold is too large, most changes from frame to frame are ignored, TTC results increase leading to wrong impression that foregrounds are far away.) Test results show that TTC results are relatively robust to the choices of  $E_t\_Threshold$ .

2) *TTC computation using whole images vs. segmented regions :* Besides, segmenting the interest foreground regions also helps to block the large TTC contributions of pixels from background regions. In general, results based on segmentation are more accurate than results based on full images. How much segmentation helps to improve the accuracy of TTC results for our test cases will be further discussed in section IV-D.

### C. TTC computation using multiple subsampling scales

In our implementation, images were subsampled after approximate low-pass filtering. Block averaging was used as a computationally cheap approximation to low pass filtering. The size of block averaging is defined as the subsampling rate which is 1 when there is no subsampling.

We had compared TTC results with different subsampling parameters based on four discussed computational models for two continuous frames. The corresponding results without subsampling are significantly larger than the results with subsampling, which somehow explains the necessity of applying subsampling technique.

The dependency of TTC estimation accuracy on subsampling rates is affected by the TTC values which will be illustrated with more details when we analyze TTC estimation results for test sequences in experiment section IV-C. As shown in figure (7) and figure (9), when the subsampling

rate is small, the estimation error for small TTCs shows the tendency of going up at the very end of the sequence because of de-focus and the large motions between frames. When the subsampling rate is high, the estimation is not reliable for large TTCs because of limited pixel input for small foreground areas and large subsampling rates. When both subsampling rate and TTC values are not too large and too small, TTC results remain relatively insensitive to the changes of subsampling rates.

### D. Hierarchical Time-to-Contact Estimation based on Fusion of Multi-Scaled Subsampling

Because of the sensitivity of TTC estimation on subsampling rate and computational models, it will help if we fuse the TTC estimation data with different computational models and at different subsampling rate. Applying TTC fusion scheme based on multi-scale subsampling helps to improve the robustness and reliability. In our experiment section, fusion scheme based on simple minimization significantly improves the estimation accuracy and reliability as in figure (7) and (9).

## IV. EXPERIMENTS AND DISCUSSION

The actual relative motion between the camera and relative to the object being imaged must be known for evaluation purpose. Thus, we created test sequences using the stop motion method, which is to move either the camera or an object in regular steps. Sequences involving camera motion are subject to small camera rotations that are hard to avoid and even small camera rotations cause significant apparent lateral image motions. In order to create reasonable stop motion sequences, we choose to move objects than to move camera. We put a toy car on the platform of a scaled optical bench with length 550mm and moved it along the optical bench by rotating a knob. The initial position of the platform is at the far end of the bench. Then for each step we slowly moved the toy car on the sliding platform toward the camera by 5mm and took a picture until the object could not be further moved because of the mechanical limitation. The motion increment is 5 mm in the depth for each frame. The accuracy of each movement is 0.1mm. The optical bench ensures the 2% = (0.1/5) accuracy in the effective TTC. Stop motion sequences produced with ordinary digital cameras suffer from the effects of automatic focus and automatic exposure adjustments, as well as artifact introduced by image compression. Thus, we chose Canon Digital Rebel xTi (EOS-400D), SLR Digital Camera Kit w/ Canon 18-55mm EF-S Lens. We took advantage of camera's remote control function and took pictures by computer operation instead of physically pressing a button on the camera to take pictures. We chose manual focus option and all parameters were manually controlled and fixed during the whole imaging process. We set small aperture and long focal length to ensure large depth of field.

Here below we first apply four different computational models to estimate TTC values based on full images as well as segmented regions for two test cases for which the relative motion is respectively along and off the optical axis in section IV-A and in section IV-B. The fusion-based

TTC estimation was applied to four stop-motion sequences whose translational motion is off the optical axis and a real video sequence as in section IV-C. The stop-motion sequences include one side-view sequence as shown in figure (5), and three front-view sequences in figure (8) that are different in the angle between the optical axis and the relative motion. Real video sequences shown in figure (10) is taken with a camera mounted on an automobile driven around Cambridge, Massachusetts. The data has been provided by the test vehicles of DARPA challenges competition.

#### A. Direct TTC computation for sequence whose translational motion is along the optical axis

The first test sequence is for translational motion along the optical axis. Figure (3) are the sample frames @ sequence 1, 16, 31, 46, 61, 76 from total 81 frames.

We compute TTC results based on both the whole images and the labeled regions in figure (3) at subsampling rate  $2 \times 2$  according to four different models as shown in figure (4). The dashed black line shows the theoretical truth value. The Cyan lines are the results based on segmented regions.

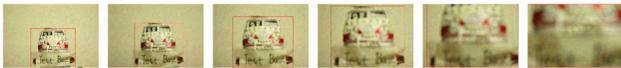


Fig. 3. Sample frames and their segmentation for a stop-motion image sequence, newCam\_Bus\_Front\_5mm (81 frames). Frame seq: 1, 16, 31, 46, 61, 76.

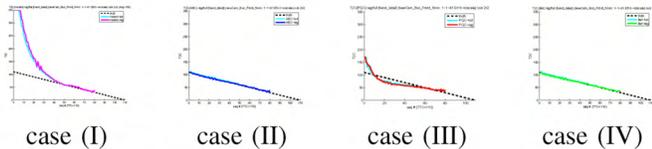


Fig. 4. The comparison of TTC computation based on the whole image and the segmented area for different cases at subsampling rate  $2 \times 2$  for sequence (newCam\_Bus\_Front\_5mm). The horizontal axis shows the frame number; the vertical axis shows the TTC. TTC thin lines: based on the whole image. TTC thick line: based on segmented areas. Dashed black line: TTC Truth value.

Among four models, the case (II) and case (IV) produce very accurate results, no matter when we use the whole image or the segmented regions. Iteration in case (IV) takes around one to five cycle before results converge. The algorithm produced TTC values that dropped linearly as expected. Even the simplest model for case (I) provides ideal results when the object is getting close to the camera, which is very important since we need high accuracy of TTC estimation when TTC becomes smaller and situation becomes more dangerous.

For case (I) and (III), they show similar trend. The detail deviation of different TTC results from the truth value is listed in the table (I), including the average based on original deviation and the absolute value of deviation. The best result is for case (II) with segmented region. The average TTC estimation error based on segmented region for case (IV) is 0.61% and 3.18% (the average of error itself and its absolute value) for case (IV).

TABLE I  
TTC ESTIMATION ERROR(PERCENT) FOR SEQUENCE IN FIGURE (3)

method	full/case(I)	full/case(II)	full/case(III)	full/case(IV)
avg	53.41	1.40	-4.65	1.34
avg	54.60	2.57	13.06	2.52
method	reg/case(I)	reg/case(II)	reg/case(III)	reg/case(IV)
avg	60.44	-0.59	-5.64	-.61
avg(abs)	61.96	2.83	15.12	3.18

The comparison between TTC results based on both full images and segmented areas is shown in table (I). The TTC and its deviation results are very encouraging. The real object is not completely planar. However the estimation results based on planar-object-computation model are still very satisfactory. Results for case (II) are very ideal even if the front view of the toy bus is not completely planar and perpendicular to the optical axis (assumption for case (II)). Our proposed method for case (IV) provides the most satisfactory performance. The results for case (I) and (III) are not ideal because we do not have a systematic method to ensure that the direction of object movement on the optical bench is completely along the optical axis for our camera. It shows that TTC estimation is more robust to object orientation than moving direction.

#### B. Direct TTC computation for sequence whose translational motion is off the optical axis

The second test sequence is for translational motion in arbitrary direction, the most general situation. The sequences are the images of the side view of a toy car moving in a direction with an angle with the optical axis. Figure (5) are the sample frames @ sequence 1, 21, 41, 61, 81, 101 from total 104 frames. Figure (6) shows the TTC estimation results as subsampling rate  $2 \times 2$  according to four methods.



Fig. 5. Sample frames and their segmentation for a stop-motion image sequence, newCam\_side\_slant10\_5mm\_day\_hf (104 frames). Frame seq: 1, 21, 41, 61, 81, 101.

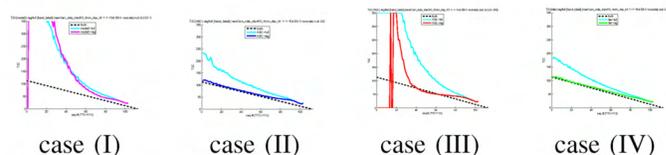


Fig. 6. The comparison of TTC computation based on the whole image and the segmented area for different cases at subsampling rate  $2 \times 2$  for sequence (newCam\_side\_slant10\_5mm\_day\_hf). The horizontal axis shows the frame number; the vertical axis shows the TTC. TTC thin lines: based on the whole image. TTC thick line: based on segmented areas. Dashed black line: TTC Truth value.

Similarly, the case (II)(IV) produce better results than for case (I)(III), and results for four cases based on segmentation are better than results based on the full image. For case (II) and (IV), results based on segmentation are quiet close to the truth

TABLE II  
TTC ESTIMATION ERROR(PERCENT) BASED ON SEGMENTATION FOR SEQUENCE IN FIGURE (5)

method	case-II sub2	case-IV sub2	reg1/fuse	reg2/fuse
avg	16.59	11.84	4.88	3.24
avg(abs)	16.59	11.96	5.21	3.96

value. The average TTC estimation error based on segmented region for case (IV) is 11.84% and 11.96% (the average of error itself and its absolute value).

### C. TTC fusion based on multiple-scale subsampling

The left and center columns in figure (7) show the TTC results based on two different segmentation schemes according to case(IV) model when we use different subsampling rates at  $1 \times 1$ ,  $2 \times 2$ ,  $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$ ,  $32 \times 32$ ,  $64 \times 64$ . The fusion result based on simple minimization operation as in section III-D is shown in the right column in figure (7). For hand-labeling segmentation scheme as shown in the top row of figure (7), the average TTC estimation error based on segmented region for case (IV) is 4.88% and 5.21% (the average of error itself and its absolute value). For auto segmentation scheme as shown in the bottom row of figure (7), the average TTC estimation error based on segmented region for case (IV) is 3.24% and 3.96% (the average of error itself and its absolute value). The details of performance evaluation in table (II) show the advantages of fusion.

We have applied three other stop-motion sequences corresponding to general situation where the relative motion are at different angle toward the cameras in order to test the robustness of the algorithm. The initial sequences are shown in figure (8) while their fusion results are shown in figure (9). TTC estimation for the side-view sequence is more accurate than for three front-view sequences since the side-view of toy bus has larger flat areas than for the front-view sequences.

Figure (11) shows the TTC estimation results based on multi-scale fusion according to our proposed gradient-based TTC computation (case model 2). Figure (11) (a1) and (a2) are TTC estimation at multiple subsampling rates ( $1 \times 1$ ,  $2 \times 2$ , and  $4 \times 4$ ) respectively based on the whole images and segmented regions. Figure (11)(b1) and (b2) are corresponding fusion results based on (a1) and (a2).

Our TTC results agree with visual estimates of vehicle motion and distances. The driver appeared to initially brake so as to keep the TTC more or less constant. The vehicle was then brought to a complete halt (at which point the computation of the TTC become unstable since C approached zero).

Unfortunately, the actual “ground truth” is not known in this case. To evaluate the performance of TTC estimation, we estimate TTC values based on traditional size-based method. We manually measure the sizes of objects in the images and estimate the TTC using equation (4) which are represented by green circles for comparison in figure (11).

Graphs of the TTC from our algorithm and the manually estimated TTC generally agree, although detailed comparison

is not possible because of the coarse quantization of the manual estimates. The size-based TTC estimation results are much more noisy than our estimation. The size-based TTC estimation between frames 100-200 vary significantly from our results. The difficulty with manual estimation of the TTC once again illustrates that the TTC algorithm presented here works with remarkably small image motions. These results show the efficiency of our algorithm for different setup and different lighting situations.

### D. TTC computation using whole images vs. segmented regions

Our test sequences are produced by the Canon camera with the same parameters. For sequences in figure (3), the intensities of background pixels are quiet similar, and the threshold of time derivative works very well in reducing the impact of background pixels. TTC estimation with four models based on both full images and segmentation are very similar as shown in figure (4) and table (I). But for the case of arbitrary movement in figure (5), results based on full images as shown in figure (6) are not as ideal as for the movement close to the direction of the optical axis. Segmentation helps to improve the accuracy of TTC estimation, TTC performance is not very sensitive to segmentation errors. We applied two segmentation scheme for sequence in figure (5), and their TTC estimation is quite similar as shown in figure (7) and table (II).

In the case of these real world video sequences, the TTC estimated by the algorithm generally tended to be somewhat lower than that estimated manually. This is because faster moving parts of the image corresponding to nearby objects contributed to the result. This effect is reduced through segmentation. TTC estimation based on the labeled regions are larger than based on whole image as shown in the comparison between figure (11)(b2) and (b1).

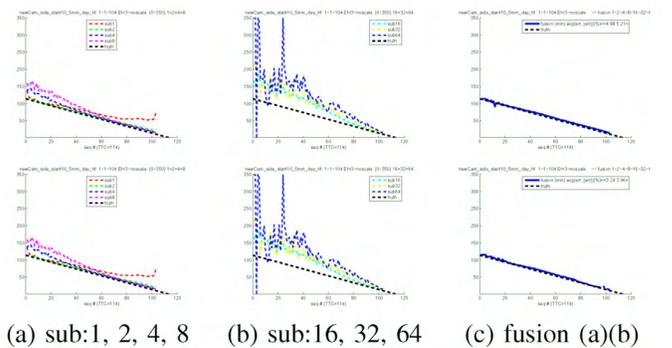


Fig. 7. The comparison of TTC fusion result with results at different subsampling rate based on given segmentation, hand labeling and auto segmentation, for sequence (newCam\_side\_slant10\_5mm\_day\_hf). The horizontal axis shows the frame number; the vertical axis shows the TTC. TTC dotted lines: results at different subsampling rates for segmented areas. TTC thick line: fusion results. Dashed black line: TTC Truth value. Top row: results for segmentation based on hand labeling. Bottom row: results for segmentation based on auto segmentation. Left/Middle column: TTC results computed with different subsampling rates. Left:  $1 \times 1$ ,  $2 \times 2$ ,  $4 \times 4$ ,  $8 \times 8$ . Middle:  $16 \times 16$ ,  $32 \times 32$ ,  $64 \times 64$ . Right column: Fusion results based on top row and center row.

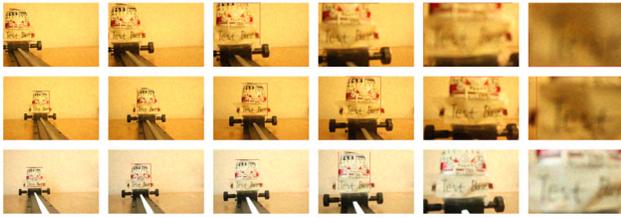
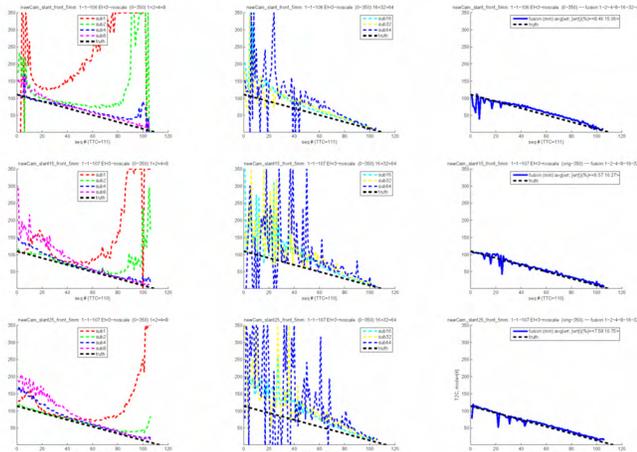


Fig. 8. Sample frames and their segmentation for stop-motion image sequences. Top row: newCam\_slant\_front\_5mm (106 frames). Frame seq: 1, 21, 41, 61, 81, 101. Middle row: newCam\_slant\_front\_5mm (106 frames). Frame seq: 1, 21, 41, 61, 81, 101. Bottom row: newCam\_slant25\_front\_5mm (107 frames). Frame seq: 1, 21, 41, 61, 81, 101.



(a) sub:1, 2, 4, 8 (b) sub:16, 32, 64 (c) fusion (a)(b)

Fig. 9. Multiscale-based TTC fusion results based on given segmentation for sequences (newCam\_slant\_front\_5mm), (newCam\_slant15\_front\_5mm), (newCam\_slant25\_front\_5mm) in figure (8). The horizontal axis shows the frame number; the vertical axis shows the TTC. TTC dotted lines: results at different subsampling rate for segmented areas. TTC thick line: fusion results. Dashed black line: TTC Truth value. (a)(b) TTC results computed with different subsampling rates. (a)  $1 \times 1$ ,  $2 \times 2$ ,  $4 \times 4$ ,  $8 \times 8$ . (b)  $16 \times 16$ ,  $32 \times 32$ ,  $64 \times 64$ . Dashed black line: TTC Truth value. (c) Fusion results based on (a) and (b). The three rows respectively correspond to three sequences in figure (8).



Fig. 10. Sample frames and their segmentation for a stop-motion image sequence, camera3CUT1-horn(93 frames). Frame seq: 100, 159, 218, 277, 336, 395.

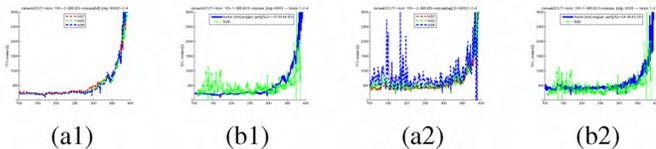


Fig. 11. The comparison of TTC fusion results and traditional size-based TTC estimation (camera3CUT1-horn). The horizontal axis shows the frame number; the vertical axis shows the TTC. (a1)(a2) TTC dotted lines: TTC results computed at subsampling rates  $1 \times 1$ ,  $2 \times 2$ ,  $4 \times 4$ ,  $8 \times 8$ . (a1) Results based on full images. (a2) Results based on labeled regions. (b1)(b2) Blue lines: Fusion results based on (a1)(a2) respectively. Green circle: size-based TTC measurement.

## V. CONCLUSIONS

We have proposed a method to determine TTC using time-varying images, which can also be used in advanced automation, automated assembly and robotics, where parts need to be moved rapidly into close physical alignment while at the same time avoiding damage due to high speed impact.

Our proposed “direct method” operates directly on the spatial and temporal derivatives of brightness, does not depend on the computation of the optical flow as an intermediate result, and does not require feature detection/tracking.

For situations with specific motion, TTC can be estimated based on the full images without any segmentation. In practice, some form of image segmentation may be useful in suppressing contributions from image regions moving in ways different from those of the object of interest. For applications with general relative motion, the final performance is robust to segmentation error. The multi-scale based fusion scheme increases the robustness to measurement error and parameter choices. In sum, our proposed method has low latency, avoids the computational load of calibration, and significantly improves the accuracy and robustness of TTC estimation.

## REFERENCES

- [1] F. Meyer, “Time-to-collision from First-Order Models of the Motion Field”, *IEEE Transaction on Robotics and Automation*, Vol.10, 1994, 792-798.
- [2] F. Meyer, P. Bouthemy, “Estimation of Time-to-collision Maps from First Order Motion Models and Normal Flows”, *ICPR1992*, 78-82.
- [3] B.K.P. Horn and S. Negahdaripour. *Direct Passive Navigation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, No.1, January 1987, pp. 168-176.
- [4] B.K.P. Horn and E.J. Weldon. Jr. *Direct Methods for Recovering Motion*, International Journal of Computer Vision, Vol. 2, No. 1, Jun. 1988, pp. 51-76.
- [5] S. Negahdaripour and B.K.P. Horn. *A Direct Method for Locating the Focus of Expansion*, Computer Vision, Graphics and Image Processing, Vol. 46, No. 3, June 1989, pp. 303-326.
- [6] B.K.P. Horn. Parallel Analog Networks for Machine Vision, in *Artificial Intelligence at MIT: Expanding Frontiers*, edited by Patrick H. Winston and Sarah A. Shellard, MIT Press, Vol. 2, pp. 531-573, 1990.
- [7] I.S. McQuirk and B.K.P. Horn and H.-S. Lee and J.L. Wyatt. *Estimating the Focus of Expansion in Analog VLSI*, International Journal of Computer Vision, Vol. 28, No. 3, 1998, pp. 261-277.
- [8] E. De Micheli and V. Torre and S. Uras. *The Accuracy of the Computation of Optical Flow and of the Recovery of Motion Parameters*, IEEE Transactions on PAMI, Vol. 15, No. 5, May 1993, pp. 434-447.
- [9] T.A. Camus. *Calculating Time-to-Contact Using Real-Time Quantized Optical Flow*, Max-Planck-Institut fur Biologische Kybernetik, Technical Report No.14, February, 1995.
- [10] P. Guermeur and E. Pissaloux. *A Qualitative Image Reconstruction from an Axial Image Sequence*, 30th Applied Imagery Pattern Recognition Workshop, AIPR 2001, IEEE Computer Society, pp. 175181.
- [11] S. Lakshmanan and N. Ramarathnam and T.B.D. Yeo. *A Side Collision Awareness Method*, IEEE Intelligent Vehicle Symposium 2002, Vol. 2, pp. 640645, 1721 June 2002.
- [12] H. Hecht and G.J.P. Savelsbergh (eds.). *Time-To-Contact*, Elsevier, Advances in Psychophysics, 2004.
- [13] Berthold Horn, Yajun Fang, Ichiro Masaki, “time-to-contact Relative to a Planar Surface”, *Proceedings of IEEE Intelligent Vehicles Symposium*, 2007.
- [14] Yajun Fang, Berthold Horn, Ichiro Masaki, “Systematic information fusion methodology for static and dynamic obstacle detection in ITS,” *15th World Congress On ITS*, 2008