A Comparison of Three Methods for Measure of Time to Contact

Guillem Alenyà, Amaury Nègre and James L. Crowley

***Abstract*— Time to Contact (TTC) is a biologically inspired method for obstacle detection and reactive control of motion that does not require scene reconstruction or 3D depth estimation. Estimating TTC is difficult because it requires a stable and reliable estimate of the rate of change of distance between image features. In this paper we propose a new method to measure time to contact, Active Contour Affine Scale (ACAS). We experimentally and analytically compare ACAS with two other recently proposed methods: Scale Invariant Ridge Segments (SIRS), and Image Brightness Derivatives (IBD). Our results show that ACAS provides a more accurate estimation of TTC when the image flow may be approximated by an affine transformation, while SIRS provides an estimate that is generally valid, but may not always be as accurate as ACAS, and IBD systematically over-estimate time to contact.**

I. INTRODUCTION

Time to Contact (TTC) can be defined as the time that an observer will take to make contact with a surface under constant relative velocity. TTC can be estimated as the distance between two image points divided by the rate of change in that distance. The result is a form of relative distance to the object in temporal units that does not require camera calibration, 3D reconstruction or depth estimation. As such, TTC can potentially provide the basis for fast visual reflexes for obstacle avoidance and local navigation.

There is a biological evidence that something like TTC is used in biological vision systems, including the human visual system, for obstacle avoidance, manipulation and navigational tasks. For the human retina, it is possible to show that the accuracy of the estimation of TTC is influenced by the relative angle of approach. TTC is more accurate when the observed motion is in the center of the retina, as well as when the distance between the observer and the obstacle is small. For artificial systems, if we know the range of translational velocities for the observing system we can choose the appropriate focal length to adapt the resolution to the expected mean speed.

It is well known that the egomotion of a robot and its relative position with respect to the obstacle cannot be estimated with a single uncalibrated camera. Part of the attraction of TTC is that the calculation relies only on image measurements and does not require camera calibration or knowledge of the structure of the environment or the size of shape obstacles. Moreover, TTC naturally encodes the dynamics of the motion of the observer. As a consequence, TTC can be used to construct motion reflexes for collision avoidance, provided that a fast, reliable measure can be made of distance in the image.

Providing a fast, reliable distance measurement for TTC is a challenging task. Classical methods to compute TTC rely on the estimation of optical flow and its first derivative [1], [2]. However, optical flow methods are iterative and tend to be computationally expensive and relatively imprecise. Calculating the derivative of optical flow to estimate TTC further amplifies noise, generally leading to an unstable and unreliable estimate of TTC. Most demonstrations of this approach tend to use highly textured objects in order to obtain a dense velocity fields [3]. Such textured objects may provide a useful laboratory demonstration, but are not generally representative of the objects observed in real world scenes.

The use of the temporal derivative of the area of a closed active contour [4] has been proposed to avoid the problems associated with the computation of image velocity fields and their derivatives. This is an additional step in the tracking of active contours that can be avoided using the parameters of the deformation of the active contour [5]. Active contour initialization is usually performed manually, and is thus difficult to implement in real moving robots.

Horn [6] has proposed a method that avoids the problem of background segmentation by computing the derivatives over the entire image. TTC obtained with this method is only valid when a large fraction of the image corresponds to the obstacle.

Some of these approaches restrict viewing conditions or allowed motions. When affine camera models are assumed [7], [4], [3], then affine image conditions are required[1]. Camera motion is sometimes restricted to planar motion [3], [8] or to not include vertical displacements [7] or cyclotorsion [8] An alternative approach is to compute TTC from scaled depth [1], [5]. This approach [9] is more complex and can be shown to introduce new constraints and additional errors when these constrains are not fully satisfied.

In this paper we propose a new method to compute TTC based on tracking active contours. We refer to this method as Active Contour Affine Scale (ACAS). We compare this method with two other recently proposed methods: Scale Invariant Ridge Segments (SIRS) [10], and Image Brightness

G. Alenyà was supported by the CSIC under a Jae-Doc Fellowship. This work was partially supported by Generalitat de Catalunya through a BE grant and by the Spanish Ministry of Science and Innovation under project DPI2008-06022

G. Alenyà is with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Llorens i Artigas 4-6, 08028 Barcelona. galenya@iri.upc.edu

A. Nègre and J. L. Crowley are at INRIA Grenoble Rhône-Alpes Research Centre 655 Av. de l'Europe, Montbonnot, France. {Amaury.Negre,James.Crowley}@inrialpes.fr

[1] In the TTC literature this condition often appears simplified as restricting the field of view (FOV), but affine viewing conditions implies also the *small depth relief* condition

Derivative (IBD). SIRS estimates TTC from the change in scale of a scale-normalised ridge segment. The IBD method avoids the segmentation and tracking of obstacles by using the difference of global brightness of the image. These methods will be introduced and discussed in more detail in the following sections.

The remainder of this article is structured as follows. In Section II the computation of TTC is introduced. Next, in Section III we introduce briefly the three methods that we will compare. Experiments are presented in Section IV, including some theoretical discussion and different practical experiences. Finally, Section V is devoted to conclusions.

II. Time-To-Contact

The time to contact is usually expressed in terms of the speed and the distance of the considered obstacle. The classical equation to compute the TTC is

$$\tau = -\frac{Z}{\frac{dZ}{dt}}\,, \tag{1}$$

where Z is the distance between the camera and the obstacle, and $\frac{dZ}{dt}$ is the velocity of the camera with respect to the obstacle. However, with a monocular camera the distance Z is generally unknown. It's possible to derive (1) by using a characteristic size of the obstacle in the image [10] and by using that the obstacle is planar and parallel to the image plane (i.e. affine image conditions)

$$\tau = \frac{\sigma}{\frac{d\sigma}{dt}}\,, \tag{2}$$

where σ is the size (or the scale) of the object in the image and $\frac{d\sigma}{dt}$ the time derivative of this scale. This equation is more appropriate as the size can be obtained directly in the image space. This reformulates the problem as a problem of estimating the size obstacle size, as well as the rate of change of size.

Note that the TTC does not rely on the absolute size of the object in the image sequence, but on the relative change in scale from one frame to another. As a consequence, the TTC computation is not dependent on camera optics or the object size, only is dependent on the depth distance and the camera velocity.

III. Methods for TTC

A. Scale Invariant Ridge Segments

A characteristic size for an obstacle can be estimated from the characteristic scale using a normalized Laplacian scale space. Characteristic scale is computed by computing the Laplacian (or second derivative) of the image for a given pixel over a range of scales. The scale at which the Laplacian is maximized is the "characteristic scale" for that pixel. A characteristic scale can be estimated at all image points except discontinuous boundaries, where the Laplacian is zero and thus has no maximum, and will return the same value for all orientations. A change of scale in the image results in a similar change in the characteristic scale.

A similar measure can be estimated using the Hessian of the gradient. As with the Laplacian, the Hessian can be computed for a given pixel over a range of scales. The scale at which the Hessian returns a maximal value is an invariant for changes of scale and rotation.

Because the characteristic scale at each pixel varies equally with changes in image scale, characteristic scale computed from the Laplacian or the Hessian can be used to estimate TTC at (nearly) all pixels in image. However, estimating TTC from characteristic scale requires registering the images so that the rate of change in scale is measured for the same image feature.

Image registration is generally estimated by some form of tracking. A popular approach is to track interest points such as the maxima of the Laplacian, as used in the SIFT [11] detector or the maxima of the Hessian, as provided by the Harris [12] interest point detector. Unfortunately the Harris detector tends to respond edge and corner points where size is not meaningful. The SIFT detector detects scale-space maxima of the Laplacian and provides a stable estimate of scale. However, the position of SIFT interest points tends to become unstable along elongated shapes, as are common in many navigation scenes. The Scale Invariant Ridge Segment (SIRS) detector [13] extends the maximum of the Laplacian as used in the SIFT detector to detect elongated shapes.

The SIRS detector consists in maximizing a score function in the 3D segment's space. We consider a ridge segment S parameterized by two vectors:

- $\vec{c} = (c_x, c_y, c_\sigma)$: center position in the image scale-space
- $\vec{s} = (s_x, s_y, 0) = \|\vec{s}\| \cdot \vec{u}$: half-edge (vector between an extremity and the center)

Then the score function correspond to the sum of the normalized Laplacian $\nabla^2 L$ combined with a symmetry detector

$$f(S) = \int_{l=-||\vec{r}||}^{||\vec{r}||} |\nabla^2 L(\vec{c}+l\cdot\vec{u})| - |\nabla^2 L(\vec{c}+l\cdot\vec{u}) - \nabla^2 L(\vec{c}-l\cdot\vec{u})|\, dl - \alpha\cdot||\vec{r}|| \tag{3}$$

where α is a parameter that represent the minimum Laplacian value needed to detect a segment.

To speed-up the maxima search, we can compute a principal direction for any center position by using the eigen vectors of the Hessian Matrix

$$\mathcal{H} = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial xy} \\ \frac{\partial^2 f}{\partial xy} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix}.$$

Then, the score function is performed in a 4 dimensional space. A second reduction is also performed by eliminating segments where Laplacian value in the center is too low.

The algorithm is depicted in Alg. 1. Its result is illustrated on Fig. 1(a). We can see that the detected segments fit well most of visible elements and the segment's scale depends on the structure size.

For the registration, the detected segments can be tracked using a particle filter described in [13]. The tracking is done

1: Computation of first and second normalized derivatives Scale-Space
2: Elimination of edge pixel using the ratio of Laplacian and Gradient values
3: **for** each pixel **do**
4: Estimation the principal direction using the Hessian matrix
5: Calculation of the score function and the length that maximize this function
6: **end for**
7: Search of local maxima

Algorithm 1: SIRS detector

in the Scale-Space, so the scale automatically estimated at any time.

B. Active Contours Affine Scale

Under weak-perspective conditions and general motion, the deformation of a set of projected points between two different frames $\mathbf{Q}, \mathbf{Q}'$ can be parameterized with a 6dof affine transformation

$$\mathbf{Q}' - \mathbf{Q} = \mathbf{WS} \tag{4}$$

where $\mathbf{Q}$ is the vector formed with the coordinates of N_Q points, first all the x-coordinates $\mathbf{Q_x}$ and then all the y-coordinates $\mathbf{Q_y}$, and $\mathbf{W}$ is the *shape matrix*

$$\mathbf{W} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{Q^x} & \mathbf{0} & \mathbf{0} & \mathbf{Q^y} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{Q^y} & \mathbf{Q^x} & \mathbf{0} \end{bmatrix} \tag{5}$$

composed of $\mathbf{Q^x}$, $\mathbf{Q^y}$, and the N_Q-dimensional vectors $\mathbf{0} = (0, 0, \ldots, 0)^T$ and $\mathbf{1} = (1, 1, \ldots, 1)^T$, and where

$$\mathbf{S} = (t_x, t_y, M_{11} - 1, M_{22} - 1, M_{21}, M_{12}) \tag{6}$$

is the 6-dimensional *shape vector* that in fact encodes the image deformation from the first to the second view.

The scaled translation in depth $\frac{T_z}{Z_0}$ from the camera to the observed object can be obtained with

$$\frac{T_z}{Z_0} = \frac{1}{\sqrt{\lambda_1}} - 1\,, \tag{7}$$

where Z_0 is the depth in the first frame and λ_1 is the greatest eigenvalue of the 2×2 matrix $\mathbf{M}\mathbf{M}^T$ obtained multiplying $\mathbf{M}$, formed from the elements of (6) with its transpose $\mathbf{M}^T$.

If we consider that the sampling period is constant then we can use the difference between two consecutive frames $i-1, i$ as an estimation for the velocity in the change of the scale. If we define the scaled depth at frame i as

$$\sigma_i = \frac{T_z}{Z_0} + 1 \tag{8}$$

then the difference of scale in two consecutive frames is

$$\sigma_i - \sigma_{i-1} = \frac{T_{z_i} - T_{z_{i-1}}}{Z_0} \tag{9}$$

and the TTC (2) can be computed as

$$\tau = -\frac{\sigma_i}{\sigma_i - \sigma_{i-1}} = -\frac{T_{Z_i} + Z_0}{T_{Z_i} - T_{Z_{i-1}}}\,. \tag{10}$$



(a) SIRS (b) ACAS

Fig. 1. Examples of SIRS and ACAS detections. (a) Each detected segment is represented by an ellipse where the main axis represents the position of the segment and the second axis represents the scale. (b) An active contour attached to the rear car window.

Note that (8) and (10) include the unknown distance Z_0. In practice, we set $Z_0 = 1$ that will scale all the translations between 1 and 0.

When the motion is known to be restricted, an approaching trajectory can be parameterized with a reduced *shape matrix* [14]

$$\mathbf{W} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{Q^x} \\ \mathbf{0} & \mathbf{1} & \mathbf{Q^y} \end{bmatrix} \tag{11}$$

and the corresponding *shape vector*

$$\mathbf{S} = (t_x, t_y, \sigma)\ , \tag{12}$$

where σ encodes the affine scale parameter. This parameter can be used instead of the scaled translation in (10) to estimate the TTC.

As can be seen in Fig. 1(b) few control points are used to parameterize a complex contour, and with this algorithm not only the TTC but also robot egomotion can be obtained [15].

As affine imaging conditions are supposed poor TTC estimations are expected when these are not satisfied. This will happen basically when perspective effects appear in the image, due mainly to a translational motion not perpendicular to the object principal plane.

The main difficulty in this approach is in the initialization of the contour. Some automatic methods have been proposed [16], [17] but it is not easy to determine the correct number and position of the control points. Moreover, when the robot is far from the obstacle its silhouette is not well defined, and sometimes it is difficult to initialize a contour that fits properly the obstacle once the robot has approached. In such conditions the tracking is difficult and the obtained scale is expected to be of poor quality.

C. Image brightness derivative

Recently has been shown [6] that time to contact can be estimated using only spacial and temporal image brightness derivatives. This method is based on the "constant brightness assumption"

$$\frac{d}{dt}E(x, y, t) = 0 \tag{13}$$

which assumes that the brightness in the image of a point in the scene doesn't change significantly. This equation can be expanded into the well known "optical flow equation"

$$uE_x + vE_y + E_t = 0 \tag{14}$$

where $u = \frac{dx}{dt}$ and $v = \frac{dy}{dt}$ are the x and y component of the motion field in the image, while E_x, E_y and E_t are the partial brightness derivatives w.r.t x, y and t.

By using a perspective projection model, a relation can be found between the motion field and the TTC and then a direct relation between image derivatives and TTC. In the special case of translational motion along the optical axis toward a plane perpendicular to the optical axis, (14) can be reformulated as

$$\frac{xE_x + yE_y}{\tau} + E_t = 0 \tag{15}$$

or

$$\frac{G}{\tau} + E_t = 0 \tag{16}$$

where τ is the time to contact of the plane, G is the radial gradient $(xE_x + yE_y)$ and x and y are the coordinates of considering pixel (measured from the principal point).

A least square minimization method over all pixels can be used to estimate an average TTC

$$\tau = -\frac{\sum G^2}{\sum GE_t}\,. \tag{17}$$

The computed TTC with IBD is known to be biased due to the inclusion of background zones in the TTC estimation [6]. This could be avoided using an object detection method in junction with a tracking algorithm, but the use of these "higher level" processing is against the philosophy of this method that was conceived precisely to avoid feature extraction or image segmentation and tracking.

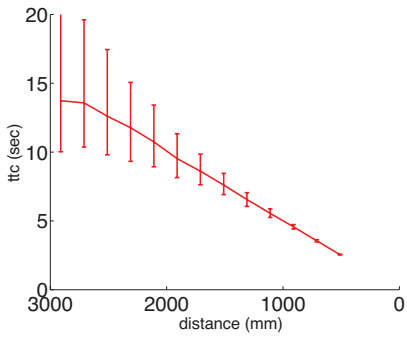
IV. Experiments

With our implementations we can process between 20 and 25 frames per second with ACAS running on a standard PC, and SIRS and IBD runing on a GPU unit.

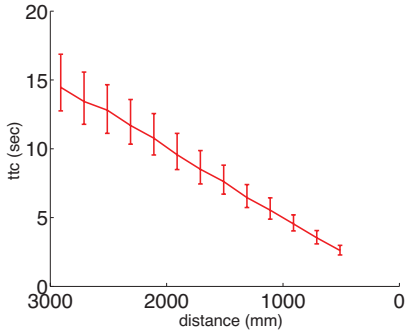
A. Completeness of the TTC

The computation of the TTC as presented is complete in the sense that a solution is always provided: in absence of noise on the inputs the scale is correctly computed (and consequently also its velocity) and the obtained TTC is the correct one; in noisy conditions, the method formulation allows to compute a valid TTC, and also reports when no motion is present and consequently when the TTC is meaningless.

We will use a simulation environment to test the behavior of the TTC formulation in presence of noise. TTC computation involves a ratio between two Gaussian variables. The result is a variable with a Cauchy distribution. As is well known it cannot be characterized as a normal distribution with the mean and variance. Instead, the median is used as it characterizes the location parameter specifying the location of the peak of the distribution. In the first experience we test the effect of noise in the scale. Noise expressed directly in the scale is difficult to determine, as it is strongly related to the image processing algorithm we will use in each case. Here we use the ACAS method (Sec. III-B) to compute the



(a) Noise in pixels

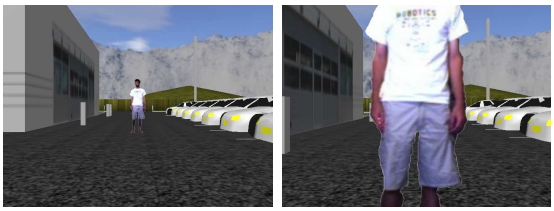


(b) Noise in framerate/velocity

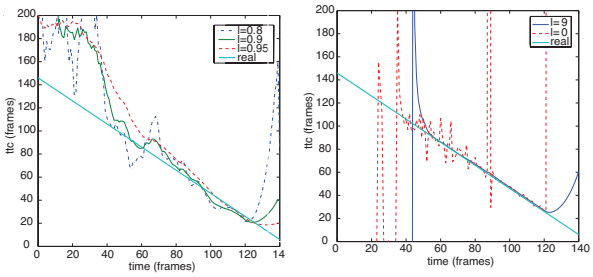
Fig. 2. Effect of noise in TTC estimation using a looming motion from 3000mm to 500mm. As expected the same level of induced noise has different effects on TTC depending on the remaining distance.

required scale. TTC has been evaluated simulating performing a Monte Carlo simulation adding Gaussian noise with zero mean and $\sigma = 0.2$ to the location in the image of the point features, with a looming motion going from 3000mm to 500mm. Results are shown in Fig. 2(a). Vertical bars in each distance represent the 25 and 75 percentile of the obtained TTC. The non-Gaussian nature of the result can be seen primarily in the beginning of the motion, between 3000mm and 1500mm, where both percentiles are different. As was expected, noise causes more error when the depth between the camera and the obstacle is large, and its effects are less important when this distance is short. This means that TTC is meaningful when we are close to the obstacle. This is useful when the robot is at relatively low velocity, but this will fail at high approaching velocities.

In the second experiment we consider the constant framerate/velocity assumption. In a real scenario, considering the framerate constant is only an approximation. In one side, it is quite difficult to obtain perfectly constant velocity with an outdoor robot, taking into account the terrain irregularities and possibly some turnings. On the other side, it is difficult to obtain a constant camera framerate when no specific hardware and software is used. In this experiment we introduced Gaussian noise with zero mean and $\sigma = 0.2$ to the derivative estimation of the scale to simulate both effects. Results are depicted in Fig. 2(b) and, as can be observed, the effect on noise in the framerate/velocity is less important than the noise in the scale.



(a) First frame (b) Last frame



(c) TTC using SIRS (d) TTC using active contours

Fig. 3. (a) Initial and (b) final images of a sequence of 140 frames involving a displacement from 8.26m to 1.21m. (c) Obtained TTC filtering with RLS algorithm. Thick line is the ground truth, and thin lines are the result applying different "forgetting" values to RLS

B. TTC in a controlled environment

We have experimented the TTC algorithms in a simulated environment. The main advantage is that ground truth here can be exactly known, and problems with illumination changes, vehicle velocity fluctuations and perturbed camera framerates can be controlled.

In the previous section we have seen the effects of noise in TTC formulation. Here we present a discussion about filtering the obtained TTC. We have used a Recursive Least Square (RLS) filter that includes a forgetting factor that allows to weight the importance of old measures.

We have also made some tests using a Kalman filter. The obtained results using Kalman filtering instead of RLS doesn't justify its use taking into account the additional complexity that is added to the algorithm and the additional work in initialize some of the required parameters.

Two frames of the simulated sequence and the results applying different forgetting values can be observed in Fig. 3. Filtering increases the stability of the computed TTC, but some inertia is introduced by the filter. This can be clearly observed at the end of the motion at frame 120 of Fig.s 3(c) and 3(d).

Comparing SIRS (Fig. 3(c)) with ACAS (Fig. 3(d)) we can see that in the beginning of the sequence ACAS is not capable of recovering precisely enough the scale (frames from 0 to 40) and TTC cannot be recovered. Conversely, SIRS can compute a value for the TTC, even if it is a little bit overestimated.

However, from frame 40 to 140 ACAS is capable of recovering the TTC, and results before filtering are clearly good enough. With these results our robot in the simulation is able to stop before crashing with the pedestrian at a security distance of 25 frames, 1 second if we assume 25fps.

C. TTC in a real scenario

To evaluate and compare different methods in the real world, we have captured a video sequence with a camera embarked on a Cycab vehicle. The speed of the vehicle is controlled and set to a constant speed during a period of ten seconds. The real TTC is not exactly know but as the speed is approximately constant, the expected TTC is linear during the constant speed period.

In these scenes, we observed a car under different points of view (Figs. 4(a), 4(b), 4(c)).

In the case of ACAS and SIRS the accuracy of the computed TTC depends on the ability to track the selected object and to measure its size in the image. The SIRS method is designed to detect and track elongated and contrasted shape, this method should thus perform well when the image contain one or more pattern with these characteristics, like the black lines on the car, or pedestrian's legs. The ACAS method can be performed only when clean contours are visible around the object's shape.

IBD method may be usable for all sequences as it only require images derivatives, neither detection nor tracking are required. Nevertheless, TTC is computed in the whole image, so it takes into account distant objects with large TTC and the obtained value is too overestimated. For this reason, we applied the algorithm only on the middle quarter of the image. As we will see, even with this modification, overestimation results are still obtained.

Figs. 4(d), 4(e) and 4(f) shows the the TTC computed by the different methods. The thin solid line is an estimation of the ground truth (the slope is obtained from the vehicle speed and the y-intercept has been manually adjusted).

In the rear and lateral car experiments (Figs. 4(a) and 4(c)) ACAS outperforms the other two methods here because the contour of the car window is clearly defined. IBD using a centered window slightly overestimates the TTC and SIRS in sequence 4(c) is not able to track correctly any elongated feature until the end of the sequence.

However, in the sequence with the car oblique (Fig.4(b)) ACAS is not able to track any contour. This is due the non frontoparallel position of the car. Perspective effects are clearly present, and in this situation affine imaging conditions are not satisfied and ACAS tracker fails to model image deformations. In that case SIRS is able to track a feature and compute an estimate of the TTC.

V. Conclusions

In this paper we have proposed a new method to calculate TTC using affine scale computed from active contours (ACAS). We have compared this method with two recently proposed measures: Scale Invariant Ridge Segments (SIRS), and Image Brightness Derivatives (IBD). Our results show that ACAS provides a more accurate estimation of TTC when the image flow may be approximated by an affine transformation, while IBD systematically over-estimate time to contact, and SIRS provides an estimate that is generally valid, but may not always be as accurate as ACAS.

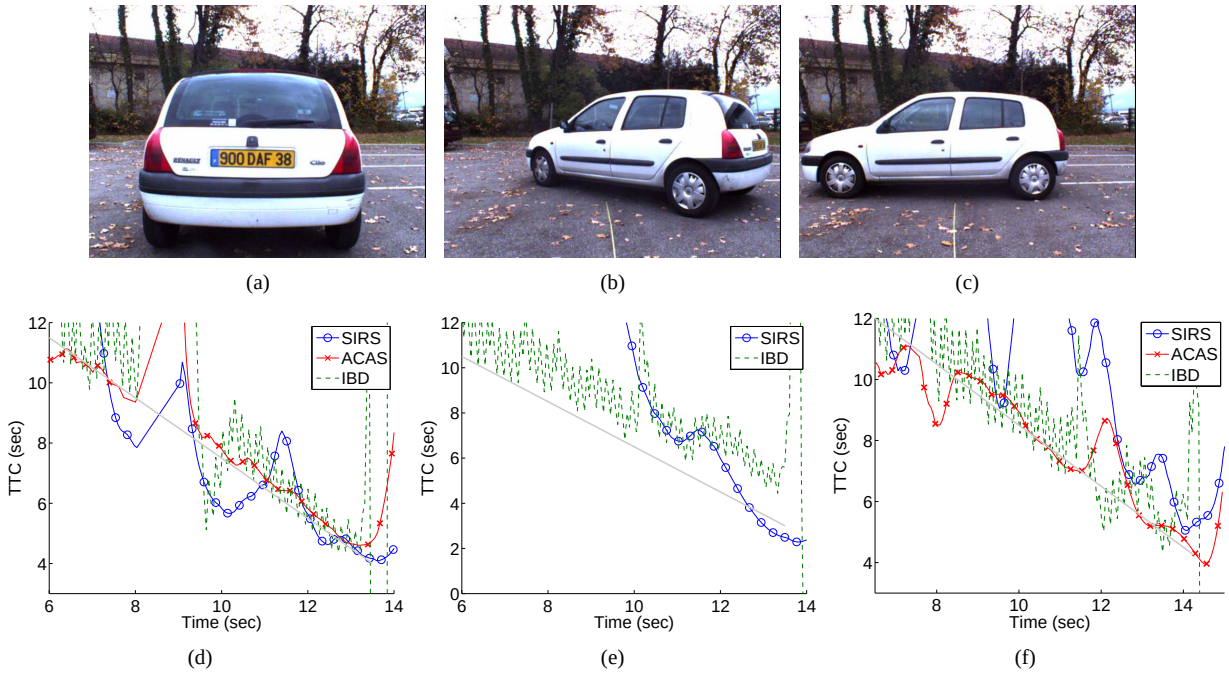


Fig. 4. Three of the experiments performed with car like obstacles. (a)(b)(c) One of the frames of the sequence. Robot motion starts aproximately at 7m and approaches up to 2m.

A weakness of the ACAS method is that it requires that obstacles first be segmented from the background. We have tested with different methods for automatic initialization of active contours, but not yet found a satisfactory one.

SIRS provides a potential means to initialize ACAS. Multiple ridges can be tracked in real time, and ridges resulting in small TTC can be flagged for more accurate computation of TTC using ACAS. Thus, it should be possible to SIRS to detect and initialize potential obstacles, and then using an affine scale computed from active contours to obtain a more accurate estimate from the obstacle.

We note that ACAS makes a strong assumption that the obstacle is viewed under affine viewing condition. While SIRS does not make an explicit assumption of affine viewing conditions, its use for TTC does rely on an implicit assumption. The accuracy of TTC measured with SIRS will degrade when perspective effects increase. We can, however, define methods to test the frontal-parallel condition to detect situations where perspective effects may degrade TTC [4].

We note that TTC can be improved by use of smoothing over time using a smoothing function such as a Kalman filter, or other methods. We have found that RLS with a forgetting parameter provide reasonably good results. However, such filtering can introduce a delay in TTC estimation, that may be a problem in real time navigation.

References

[1] M. Tistarelli and G.Sandini, "On the advantadge of polar and log-polar mapping for direct estimation of time-to-impact from optical flow," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, no. 4, pp. 401–411, 1993.

[2] N.Ancona and T.Poggio, "Optical flow from 1d correlation: Aplication to a simple time-to-crash detector," *Int. J. Comput. Vision*, vol. 14, no. 2, pp. 131–146, 1995.

[3] J. Santos-Victor and G. Sandini, "Visual behaviors for docking," *Comput. Vis. Image Und.*, vol. 67, no. 3, pp. 223–238, 1997.

[4] R.Cipolla and A.Blake, "Surface orientation and time to contact from divergence and deformation," in *Proc. 4th European Conf. Comput. Vision*, 1992, pp. 187–202.

[5] E. Martinez, "Recovery of 3D structure and motion from the deformation of an active contour in a sequence of monocular images," Ph.D. dissertation, Universitat Ramon Llull, 2000.

[6] B. Horn, Y. Fang, and I. Masaki, "Time to contact relative to a planar surface," in *Intelligent Vehicles Symposium*, 2007, pp. 68–74.

[7] F.G.Meyer, "Time-to-collision from first-order models of the motion field," *IEEE Trans. Robot. Automat.*, vol. 10, no. 6, pp. 792–798, 1994.

[8] M. Lourakis and S. Orphanoudakis, "Using planar parallax to estimate the time-to-contact," in *Proc. 13th IEEE Conf. Comput. Vision Pattern Recog.*, vol. 2, Fort Collins, Jun. 1999, pp. 640–645.

[9] C. Colombo and A. Del Bimbo, "Generalized bounds for time to collision from first-order image motion," in *Proc. IEEE Int. Conf. Comput. Vision*, Corfu, Sep. 1999, pp. 220–226.

[10] A. Nègre, C. Braillon, J. L. Crowley, and C. Laugier, "Real-time time-to-collision from variation of intrinsic scale," in *Proc. Int. Symp. Experimental Robotics*, Rio de Janeiro, Jul. 2006, pp. 75–84.

[11] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] C. G. Harris and M. Stephens, "A combined corner edge detector," in *Proc. Alvey Vision Conf.*, Manchester, Aug. 1988, pp. 189–192.

[13] A. Nègre, J. L. Crowley, and C. Laugier, "Scale invariant detection and tracking of elongated structures," in *Proc. Int. Symp. Experimental Robotics*, Athens, Jul. 2008.

[14] G. Alenyà and C. Torras, "Depth from the visual motion of a planar target induced by zooming," in *Proc. IEEE Int. Conf. Robot. Automat.*, Rome, Apr. 2007, pp. 4727–4732.

[15] G. Alenyà, E. Martínez, and C. Torras, "Fusing visual and inertial sensing to recover robot egomotion," *Journal of Robotic Systems*, vol. 21, no. 1, pp. 23–32, 2004.

[16] T. Cham and R. Cipolla, "Automated b-spline curve representation incorporating mdl and error-minimizing control point insertion strategies," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, no. 1, 1999.

[17] H. Yang, W. Wang, and J. Sun, "Control point adjustment for b-spline curve approximation," *Computer-Aided Design*, vol. 36, no. 7, pp. 639–652, 2004.