2004 IEEE Intelligent Vehicles Symposium
University of Parma
Parma, Italy • June 14-17, 2004

Stereo-Based Vision System for Automotive Imminent Collision Detection

Peng Chang Theodore Camus Robert Mandelbaum
Sarnoff Corporation
Princeton, NJ 08543
{pchang, tcamus, rmandelbaum}@sarnoff.com

Abstract

Imminent collision detection is an important functionality in the area of automotive safety. In the event that an unavoidable collision can be detected in advance of the actual impact, various measures can be taken to mitigate injury and damage. In this paper, we demonstrate that stereo vision is a promising solution to this problem. Our prototype system has been rigorously tested for different colliding scenarios (e.g., different intersection angles and different travelling speeds), including live tests in an industrial crash-test facility. We explain the novel algorithms behind the system, including an algorithm for detecting objects in depth images, and algorithms for estimating the travelling velocity of detected vehicles. Quantitative results and representative examples are also included.

I. Introduction

We are teaming up with our partners in the automotive industry to develop vision based prototype systems for imminent collision detection. Our goal is to develop a system, in cooperation with our industry partners, to reliably detect a potential collision with sufficient warning time to actuate appropriate safety measures and mitigate damage or injury resulting from the collision.

Two key tasks in collision detection are to detect the colliding vehicle and to accurately estimate its relative motion. Compared to other sensors, such as radar, vision system has some unique advantages, such as the inherent high spatial resolution of targets, which can provide high confidence of the threat identity. However, detecting colliding vehicles and accurately estimating their motion with stereo vision are challenging tasks, since the depth images from stereo often exhibit substantial noise and outliers. Furthermore the depth images are often sparse, containing "holes", due to a lack of textures in the original images. Therefore successful algorithms have to be both efficient and robust to such artifacts.

In the following sections, we explain in detail the algorithms we developed to perform object detection in depth images and robust velocity estimation for detected vehicles. Finally, promising results are presented.

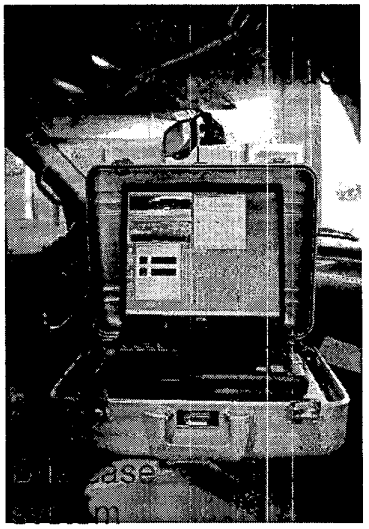

Fig. 1. Hardware setup of the collision detection system

II. System Description

Figure 1 shows the complete hardware setup for the prototype system. A lightweight stereo head is mounted on the windshield and is connected to a portable "briefcase" system. Inside the briefcase is a laptop equipped with a commercially-available *Acadia* stereo board, which can compute disparities from the stereo images at video rates (30Hz). The laptop provides the computing power for collision detection and other algorithms. The architecture and performance of the *Acadia* chip can be found in [8] . Our focus is on the collision detection algorithms in this paper.

The basic requirement of the imminent collision detection system is to detect the threat vehicle, that is going to collide with the host vehicle, and predict the collision time and the impact point as accurately as possible. This information is sufficient to control the actuation of an appropriate active safety system.

In our system, these requirements are achieved by the following steps:

1) Detect possible imminent collision threats upon their presence in the stereo depth image

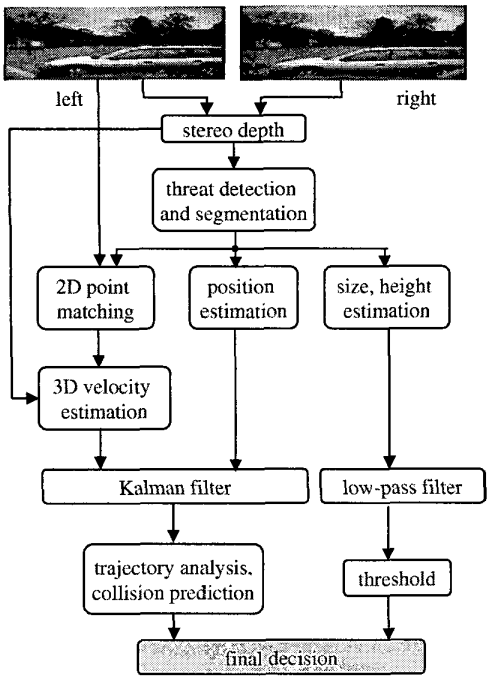

Fig. 2. Overall flowchart of the vision system

2) Estimate relevant properties of the threat, such as size, height, width etc.
3) Estimate the motion of the threat
4) Predict the threat car trajectory and the potential colliding spot on the threat car

Step 1 is in fact a segmentation problem in the stereo depth images. Reliably segmenting threat cars under varied illumination and reflection conditions is nontrivial. Details about the threat detection algorithm is in Section II-A. Step 2 is important because it provides additional confidence in the threat identity. Given proper segmentation, Step 2 can be straightforward from the depth measurements. For Step 3 a novel algorithm is developed to reliably estimate the motion of the threat object by combining both 2D and 3D information. The details of the motion estimation algorithm is described in Section II-B. In the current implementation, Step 4 is achieved by assuming a constant velocity model in the trajectory prediction. Kalman filtering is used to compensate for the noise in the velocity measurements.

Figure 2 shows the flowchart of the vision algorithms. In the following sections, we explain more details about colliding vehicle detection and velocity estimation.

A. Threat detection and segmentation

The task of threat detection is to segment the threat car or other object from the depth images. Previous work on stereo based segmentation either assume static backgrounds [6] [3] or use relatively simple grouping rule [9]. For our system, the host vehicles usually move in high speed and under a variety of lighting conditions. In addition, stereo depth images are usually sparse on car bodies, due to shiny metallic surfaces lacking texture. Therefore, a reliable segmentation is more challenging.

We adopt a bottom-up approach, which is often used in low level vision after [7]. At first the input depth image is represented with primitives, namely a grid of planar patches, and then each patch is labelled as predefined types according to its position and normal vector. Then, a grouping algorithm is used to group the small patches together according to a certain syntactic grammar. The output of the grouping algorithm actually forms the representation for the foreground object. In practice we find that this method works very well on the often noisy and sparse depth data. Figure 3 shows the overall flowchart of the detection and segmentation process.

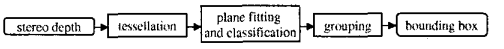

Fig. 3. Overall flowchart of detection and segmentation

There are advantages of using small patches or tiles as the intermediate representation, instead of using the 3D points in depth images directly. By incorporating all the 3D points within the specified region, the patch is a more robust representation than the individual 3D points. In addition, by reducing the depth images to a patch grid representation, the computation can be made more efficient.

To compute the patch grid, each depth image is first tessellated into a grid of tiles. For each tile, a planar patch is fit through the points within it. Singular value decomposition (SVD) can be applied to the plane fitting process. Let the set of 3D points within a patch be $\{(x_i, y_i, z_i) \| i = 1...N\}$, and X is the N by 3 matrix by stacking $\{(x_i, y_i, z_i) \| i = 1...N\}$ together. It can be shown the normal vector of the best fitting plane is given the eigen vector corresponding to the smallest eigen value of matrix A, where

$$A = (X - M)^T (X - M)$$

and M is a matrix containing the mean of (x_i, y_i, z_i) at each row. Fast implementations of the SVD algorithm exists for positive semi-definite matrixes, which is the case for the matrix A.

Once we have a primitive representation for the scene, in this case, the patch grid, a grouping step can be taken to abstract the desired object from it. As we noted before, the effect of specularities and lack of texture, among other

factors, mean the stereo depth images are often sparse and error prone. Therefore, naive grouping rules based on rigid distance are likely to fail. In our system, a syntactic based grouping method is used. Examples of using syntactic grammar on 2D pattern classification can be found in [5]. Here we apply the 2D syntactic grammar for grouping purposes.

Once the planar patch representation is available, the patch is first labelled as one of several predefined types, according to its position and normal direction. We list here the relevant patch types for car detection.

- a $CarSide$ patch, if it is high enough and with an almost horizontal normal
- a $CarTop$ patch, if it is high enough and with an almost vertical normal
- a $Boundary$ patch, if it is close to ground but with an almost horizontal normal, or if it is higher than ground but lower than top patch and with an almost vertical normal

As we can see the labelling is based on the position and the normal. Since there is always noise in the stereo computation, using either one of them alone leads to inferior results. The actual thresholds depend on the calibration parameters and the targets of interest as well. It is worth noting that, in most cases, patches labelled as the side of a car or as the top of a car often correspond to part of an obstructing target in front of the host car. The boundary patches are often patches containing mixed parts of the target and background.

Next, we can group the patches together according to some predefined syntactic pattern. The semantics of the patch grouping for an object can be expressed by a grouping grammar. We list an example grammar for a car as follows.

- $< CarBody >::= < CarSide >$
- $< CarBody >::= < CarBody >$ BESIDE $< CarSide >$
- $< Car >::= < CarBody >$ BESIDE $< Boundary >$
- $< Car >::= < Boundary >$ BESIDE $< CarBody >$
- $< Car >::= < CarTop >$ ON TOP OF $< Car >$

The grammar usually conveys how the final segmentation (a bounding box) can be composed from the patches inside it. By using a syntactic approach, we are able to group sparse patches, which cannot be grouped if only a rigid distance based grouping rule is deployed. The grouping process can iteratively group the labelled patches into one object according the grammar.

Figure 4 shows a couple examples of the grouping result on synthetic data. Red rectangles are the output bounding boxes. Green rectangles are the patches that are grouped together to form the bounding box. Notice that the patches

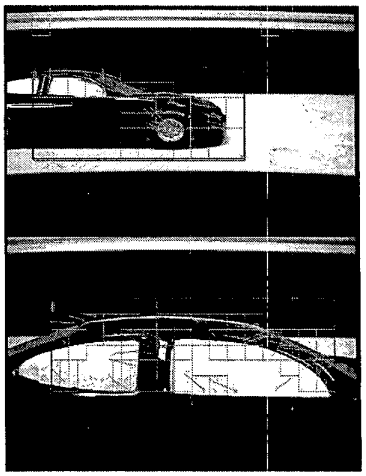

Fig. 4. Typical segmentation result from synthetic data

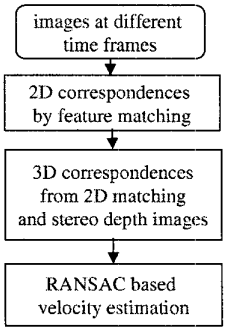

Fig. 5. Flowchart for velocity estimation

are sparsely distributed due to the sparse depth information from the stereo. The arrows will be explained in later section. The two images correspond to frames when the target first enters the field of view and the collision with the host car at the end of the sequence, respectively.

The height, width, size and location of the threat object can be measured directly once the bounding box is obtained. A threat is declared only if its size, height etc. meet certain defined criteria. This step can reduce the false alarms caused by stereo noise or spurious objects not threatening to the host vehicle.

B. Velocity measurement

In order to predict if a collision is likely to happen after the threat detection, the motion of the threat has to be estimated fairly reliably. Figure 5 shows the process of our method for velocity estimation.

Our approach is to first track a set of features on 2D from the segmented car images, and then use the 2D cor-

respondences across time to find the 3D correspondences across time. From the 3D correspondences across time, the rigid motion of the threat car can then be estimated.

An efficient feature tracking algorithm has been developed to find the 2D feature correspondences across time. In the heart of the feature tracking algorithm is a Harris feature detector and a highly optimized correlation based matching. Once we have the correspondences across time for the set of 2D features, it is easy to find the 3D position correspondences across time for the same set of features, given the depth images from stereo.

Now we have two corresponding 3D point sets P_i and $Q_i, i = 1...N$, such that $Q_i = RP_i + T + V_i$, where N is the number of points in the sets, and R is a rotation matrix, T a 3D translation vector and V_i a noise term. Standard methods exist to solve for optimal R and T given $N \geq 3$. We choose the method described in [1]. A straightforward implementation of [1] leads to inferior results, largely due to the severe noise and outliers in the 3D point sets P_i and Q_i. To accommodate the severe outliers in the 3D point correspondences, a robust method is developed to estimate the velocity based on RANSAC (Random Sample Consensus), which is known to be robust against outliers [4]. The algorithm is described below.

```
      1. select k points from the
correspondence sets
      2. solve for R and T
      3. find how many points (out
of N) fit within a
         tolerance, call it M.
      4. if M/N is large enough,
accept the result and exit
      5. repeat 1..4 L times
      6. fail
```

Figure 6 shows typical tracking result from the 2D tracker both on synthetic and real data. The 2D flow draw in green shows the features being tracked by the 2D tracker. The 2D flow in red is the feature set, which are selected by the RANSAC algorithm to compute the final rigid motion. The yellow bull's eye is an indication of the colliding spot, which is explained later.

It is worth noting that it is possible to directly establish 3D correspondences directly from the depth images, with algorithms such as ICP (Iterative Closest Points) [2]. But our experiments indicate that it is computationally expensive and often not robust against the noise and outliers in stereo.

For automotive collision detection, we find it is even more reliable to assume the rigid motion to be pure translation. In fact, in the most interesting cases where two vehicles colliding at high speed, a constant translational motion is usually sufficient, and more reliable to estimate.

Fig. 6. Examples of the 2D feature tracking and motion estimation

120 degree	15 mph	22 mph	30 mph	40 mph
0 mph	701.2	736.3	499.5	367.5
15 mph	597.2	474.7	350.8	287.6
30 mph	286.5	314.8	249.6	182.5
40 mph	185.2	220.3	221.1	154.9
90 degree	15 mph	22 mph	30 mph	40 mph
0 mph	935.6	633.8	468.7	365.9
15 mph	454.1	546.8	410.3	306.1
30 mph	174.3	210.7	241.7	241.4
40 mph	138.3	140.8	141.2	141.8
60 degree	15 mph	22 mph	30 mph	40 mph
0 mph	1132.8	738.0	532.8	400.42
15 mph	600.0	625.6	508.6	354.0
30 mph	173.4	242.5	266.9	293.7
40 mph	86.1	134.7	125.3	209.2

TABLE I

SYSTEM ADVANCED WARNING TIME FOR COLLISION UNDER DIFFERENT INTERSECTION ANGLES (120, 90, 60 DEGREE) AND TRAVELLING SPEEDS, ALL TIME ENTRIES IN MILLISECONDS

The accuracy of the motion estimation can be illustrated by the accuracy of the collision prediction, which we will show in Section III.

III. EVALUATION

The algorithms described above have been systematically evaluated using toy car models and computer-controlled camera placement. Simulated intersection angles of 60, 90, and 120 degrees were tested with target speeds of 15mph, 22mph, 30mph and 40mph and host speeds of 0mph, 15mph, 30mph and 40mph. For each car model we collect a data set covering all the collision scenarios. For each collision scenario, we estimate the advanced warning time, which is the time interval between the time of detection and the time of the actual collision. Longer advanced warning time means that the system has more time to react to the upcoming collision. Table I shows the advanced warning times estimated from one data set.

The following results demonstrate the system performance in real-world data collected by the authors. Additional rigorous testing results, including crash testing

results performed together with our industry partners, have demonstrated similar performance.

The examples shown were taken with a dense video sampling rate, and sub-sampled to provide an approximately 30mph effective speed for both the target and host vehicles.

Figure 7 and Figure 8 show typically correct collision detection under different colliding scenarios. Small green boxes are patches detected as a potential threat object. Red boxes are declared threat cars. Arrows are 2D optical flow from 2D point tracking for velocity estimation. If an imminent collision is detected, a yellow bulls eye is placed at the predicted impact point.

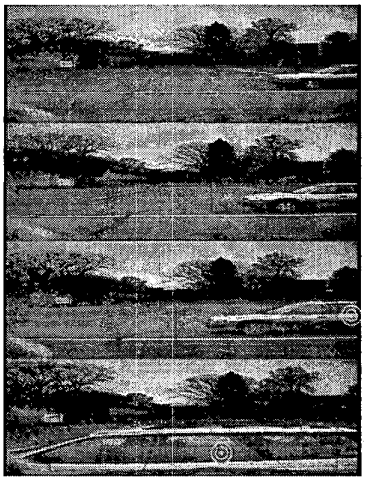

Fig. 7. Perpendicular collision: Vehicle entry, threat detection, collision determination, and final impact point.

Fig. 8. Oblique collision: Vehicle entry, threat detection, collision determination, and final impact point.

Figure 9 shows a late-miss collision scenario that might otherwise confound a collision-detection algorithm. While the target is correctly detected as a threatening object, a collision would not occur and the algorithm correctly does not detect a collision, despite the close proximity of the target vehicle.

Fig. 9. Late near-miss: Vehicle entry, threat detection, and final exit. No collision is detected.

The differences between an impact determined to be a dangerous collision and one not considered as dangerous can be subtle. Figure 10 shows an oblique head-on collision, and the algorithm correctly extrapolates the collision point. Figure 11, however, does not present a sufficiently large target to be labelled a threat.

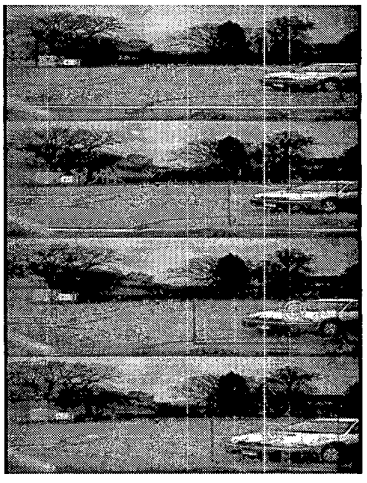

Fig. 10. Oblique hit: Vehicle entry, threat detection, collision determination, and final impact point.

While the algorithms described are optimized for side-impact collisions, they are also effective for head-on collisions. Figure 12 shows the results for a typical head-on collision.

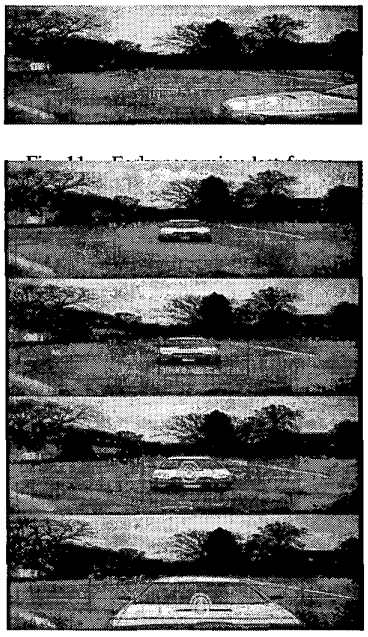

Fig. 12. Head-on collision: Vehicle entry, threat detection, collision determination, and final impact point.

IV. Conclusion

In this paper we present a system for automatic collision detection for automotive applications. The front-end of this system is video-rate real-time stereo. Novel algorithms have been developed for the task of automotive collision threat determination and collision detection. This system has been carefully evaluated by the authors and our industry partners and has demonstrated the concept feasibility of viable automotive vision system products.

Acknowledgment

The authors would like to thank Dr. John Southall, Dr. David Nister and Dr. Zhihai He for their contribution to this work.

References

[1] K.S. Arun, T.S. Huang, and S.D. Blostein. Least-square fitting of two 3d point sets. *IEEE Trans. Pattern Anal. Machine Intell.*, 9(5):698–700, 1987.

[2] P.J. Besl and N.D. McKay. A method for registration of 3d shapes. *IEEE Trans. Pattern Anal. Machine Intell.*, 14(2):239–256, 1992.

[3] C. Eveland, K. Konolige, and R.C. Bolles. Background modeling for segmentation of video-rate stereo sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1998.

[4] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24, 1981.

[5] K.S. FU. *Syntactic Pattern Recognition and Applications*. Prentice-Hall, Englewoods Cliffs, 1982.

[6] I. Haritauglu, D. Harwood, and L.S. Davis. w^4s: a real-time system for detecting and tracking people in $2\frac{1}{2}d$. In *European Conference on Computer Vision*, 1998.

[7] D. Marr. *Vision*. W.H. Freeman and Company, 1982.

[8] G. van der Wal, M. Hansen, and M. Piacentino. The acadia vision processor. In *Proc. of the IEEE Intl. Workshop on Computer Architecture for Machine Perception*, 2000.

[9] L. Zhao and C. Thorpe. Stereo and neural network based pedestrian detection. In *International Conference on Intelligent Transportation Systems*, 1999.