A Trinocular Stereo System for Highway Obstacle Detection

Todd Williamson and Charles Thorpe
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{Todd.Williamson,Charles.Thorpe}@ri.cmu.edu

Abstract

This paper presents a trinocular stereo algorithm that we have developed as part of an obstacle detection system that detects small obstacles at long range on the highway. We describe the stages of the stereo algorithm itself, and the reasons behind the design decisions that were made. This is followed by a description of the obstacle detection system that we have built. Finally, we show results from our system for a variety of different obstacles.

1. Introduction

Highways present a dynamic environment with real-time constraints. A vehicle travelling at highway speeds needs to detect obstacles at ranges of 100 meters in order to swerve or brake. Sensors such as automotive radar do not have the acuity to find small obstacles at this distance, and have significant difficulties with non-metallic obstacles such as wood, cement, or animals. While many competing methods have been proposed for on-road obstacle detection, most of the work has focused on detecting large objects, especially other vehicles (e.g. [3]). Many of these methods can successfully detect moving vehicles, but the more difficult problem of finding small, static road debris such as tires or crates remains unsolved. Although the problem of detecting static obstacles has been tackled in both the cross-country and indoor mobile robot navigation literature (e.g. [4]) these systems have operated at low speeds (5-10 mph) and short range.

At Carnegie Mellon we have developed a stereo vision system that is capable of detecting 14 centimeter tall obstacles at distances of 100 meters or more. In order to achieve this level of performance, our system relies on four key points:

1. The system is engineered such that small obstacles produce significant disparities even at 100 meters ahead. To do this, we use a set of 35 mm telephoto lenses, giving a horizontal field of view of 14° and a vertical field of view of 10°. We also use a long baseline of around 1.2 meters.

2. We use methods for easily and accurately calibrating the system, by matching images of planar surfaces. This is particularly important since a 640x240 image taken with the telephoto lenses has pixels that are 0.02° wide and 0.04° high. Ease of calibration is important because even rigid camera mounts will drift significantly over weeks or months of driving.

3. We use a well-designed stereo algorithm. The algorithm makes use of multiple baselines to reduce the number of matching errors. It also makes use of a high-gain Laplacian of Gaussian filter to enhance image texture.

4. Using projective geometry, we have a method for detecting whether a given pixel is more likely to belong to a horizontal surface or a vertical surface directly from the stereo matching results.

This paper concentrates the third point, the details of the stereo algorithm that we use, and how it can be implemented efficiently in software. A more in-depth discussion of the detection of surface orientation directly from stereo data can be found in [10]. The entire system is described in detail in [11].

2. Stereo Algorithm

The research described in this paper originated from an effort to apply the CMU Video-Rate Multibaseline Stereo Machine to the problem of detecting highway obstacles. The stereo algorithm used in this research is thus based on the algorithm used by the stereo machine.

The architecture of the stereo machine is described in some detail in [2], though that paper does not describe the reasoning that went into their choice of algorithm. Some of that reasoning will be described later in this paper.

The basic system that we have constructed is shown in Figure 1. We have a trinocular stereo rig mounted on top of the vehicle, with the longest baseline being about 1.2 meters. The third camera is displaced about 50 cm horizontally and 30 cm vertically in order to provide a short vertical baseline (the reasons for this will be explained later). The images from each of these cameras are passed through a Laplacian of Gaussian (LoG) filter and a rectification stage. Then all three images are passed into a stereo matching algorithm. The results of stereo matching are used for obstacle detection and localization.

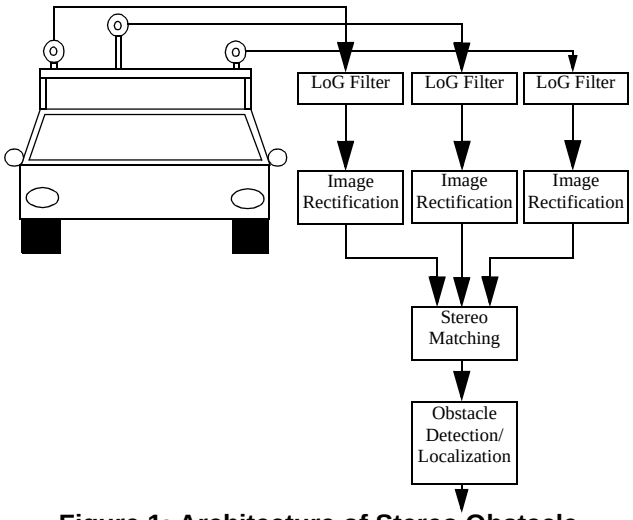

Figure 1: Architecture of Stereo Obstacle Detection System

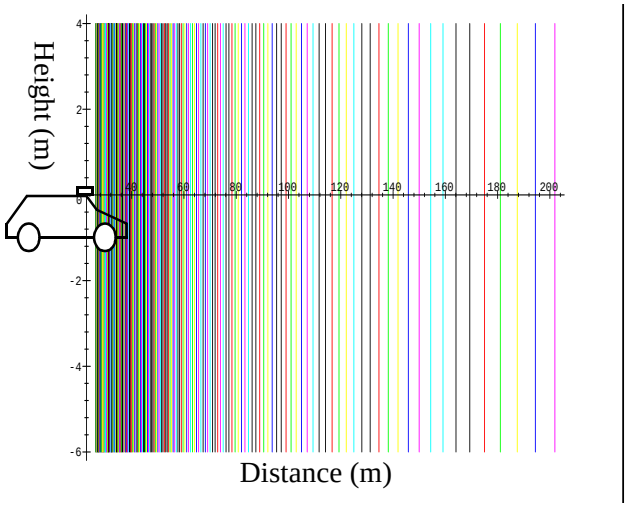

Figure 2: Planes of constant disparity for usual stereo approach

3. Ground Plane Stereo

In the most common implementation of stereo matching, an implicit assumption is made that pixels over a local region of the image all share the same distance from the camera. This occurs because the search along the epipolar line compares a square window surrounding the target pixel in one image with a square window in the other image. Thus the disparity level that matches best indicates which of a set of planar surfaces of constant depth the given pixel is likely to lie on. This set of planes is illustrated in Figure 2.

This assumption is violated when viewing a horizontal surface such as the road surface shown in Figure 4, since pixels that are higher in the image are farther away than pixels which are lower. It is a well-established result (derived in e.g. [3]) that if the surface being viewed is planar, then the relationship between disparity and image row

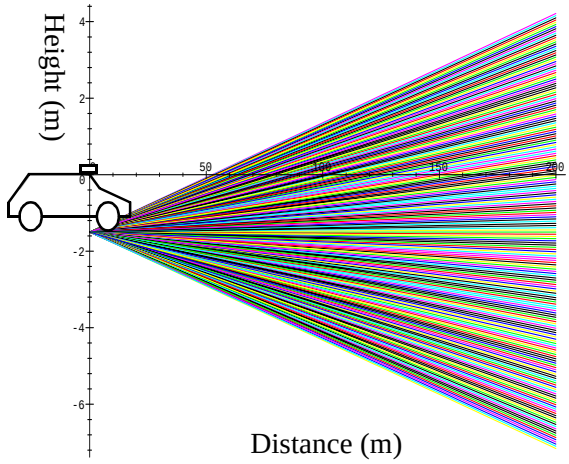

Figure 3: Planes of constant disparity for ground plane stereo

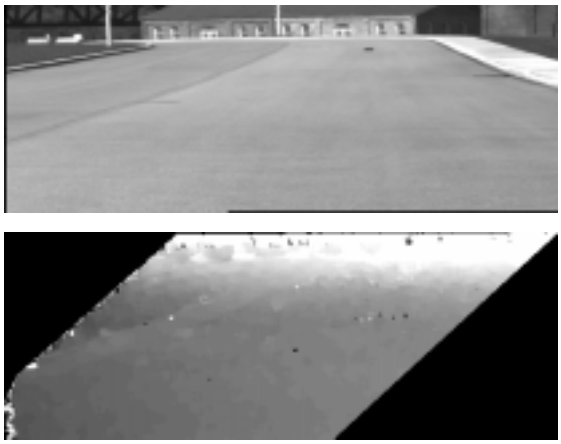

Figure 4: Example "Ground Plane Stereo" output from our system

is linear.

This result leads to the idea of compensating for the slope of the ground during the image rectification process, before performing stereo matching [1]. After performing such compensation, pixels that lie on a horizontal plane will appear in precisely the same location in both images.

If we perform normal stereo matching using images that have been compensated in this manner, the "disparity" that matches best now gives us a somewhat different result. Instead of the planes of constant depth shown in Figure 2, we are now effectively searching over the set of planes shown in Figure 3. We call this method "ground plane stereo". An example of the output of this method for a typical road scene is shown in Figure 4. The road pixels appear on a small number of planes whose values show up as medium gray. A slight upward slope from left to right is discernible. The building in the background is above the ground and therefore appears brighter.

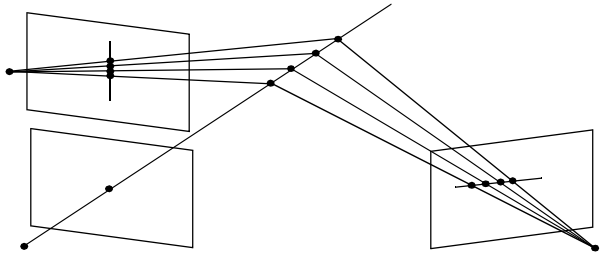

Figure 5: Three cameras in an "L" configuration give different epipolar directions

4. Multibaseline Stereo

The method for computing stereo from more than two cameras that is used in this paper was first described in [7]. The algorithm described there is called SSSD-in-inverse-distance. The main idea, that matching errors from multiple baselines can be added together to evaluate different possible geometries, has been retained in this work.

Although only two cameras are required to compute range from image data, there are several advantages to using more than two cameras for stereo vision:

1. since the epipolar direction is the same as the direction of camera displacement, it is possible to arrange for the epipolar directions of multiple cameras to lie in different directions in the image, thus taking advantage of image texture in any direction (an example of this is illustrated in Figure 5). An example of where this is useful is when viewing a horizontal feature such as the curb in Figure 4. When viewing this region with only a pair of cameras with a horizontal baseline, there is very little in the image to distinguish one location from another. The addition of a vertical baseline allows us to take advantage of the available texture.

2. repeating texture in the image can confuse a two camera system by causing matching ambiguities; these ambiguities are eliminated when additional cameras are present, assuming that the camera spacing is not an integer multiple of the texture spacing; the latter issue can be avoided by not placing the cameras at an even spacing.

3. as in any measurement process, additional measurements allow more accurate results by averaging noise; in the case of a large number of cameras, outliers can be rejected by voting or robust statistics

4. shorter baselines are less prone to matching error (since the images are more similar) while longer baselines are more accurate; the combination is better than either alone

5. different regions of space are occluded for each camera pair; therefore the problems caused by occlusion are somewhat ameliorated by using multiple cameras

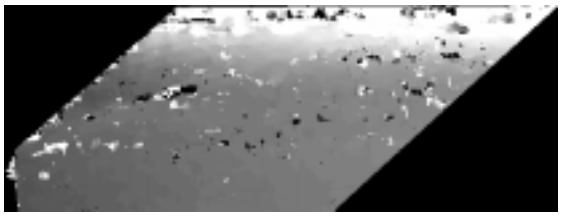

Figure 6: Output when only two cameras are used

For advantages 1 and 2 adding a third camera is sufficient; fourth and additional cameras do not yield any additional benefits. The advantages of 3-5 continue to grow past the fourth camera. Thus there is a large benefit to adding the third camera, and the benefits diminish with the fourth and additional cameras.

The importance of the third camera to our system can be seen in Figure 6, where we show what the output from Figure 4 would look like if computed with only two cameras. The amount of noise throughout the image shows a large number of pixels where two cameras are insufficient to find the correct matching result.

5. LoG Filtering

Convolution with the Laplacian of Gaussian operator has a long history as a feature detector. It was first used by Marr and Hildreth [5] as an edge detector. Nishihara [6] first used the sign of the LoG-filtered image for stereo matching. The use of more bits of information from the LoG filter was a natural extension of this.

Since the Laplacian of Gaussian is a second derivative operator, places where the LoG-filtered image is zero are places where the intensity of the original image has maximum variation, i.e., edges. In addition to being a good edge detector, the LoG filter also has the following two properties:

- it has a tunable Gaussian filter for filtering out high-frequency image noise

- since the LoG function naturally integrates to zero, any bias in intensity between the cameras is eliminated (it subtracts out)

Since the zero crossings of the LoG-filtered image are interesting points, it makes sense that points that are near to zero will also be interesting. Therefore, as a preprocessing step, we apply the filter with a high gain, and saturate values that overflow or underflow at the maximum and minimum representable values. This has the effect of accentuating regions that are near to zero crossings. We use a small standard deviation for the filter since we do not want to aggressively remove high frequency texture from the image.

Figure 7: Result of LoG filtering the top image of Figure 4

Figure 8: Example of stereo output without LoG filter

The result of LoG filtering applied to the top image of Figure 4 is shown in Figure 7. The increase in image texture, particularly on the road surface, is very apparent. In practice, the texture extracted by this method is consistent even between different cameras, and thus is very useful for stereo matching in such bland environments.

One question that remains is how large the gain on the filter should be. Experiments performed with a large number of different gains in an attempt to determine the optimal gain value had predictable results: the optimal gain depends on the relative contrast of the image. Images that have very little contrast benefit from a large gain (even if the noise is amplified greatly, it is still better than having no signal to match whatsoever). On the other hand, images with high contrast match well without any additional enhancement.

In practice, the gain should probably depend on the image data itself. A system that automatically adjusted the gain so that the contrast was as high as possible without overly saturating the image would be a good solution, though nothing of this type has been implemented in our system.

The importance of the LoG filter to our algorithm is illustrated in Figure 8. The lack of image texture on the road surface causes the entire region to be unmatchable, though regions with higher texture, such as the building in the background and the small obstacle are matched correctly.

6. Rectification and Interpolation

After calibrating the camera system, the necessary geometric constraints between cameras for stereo matching are known. For each pixel in the reference image, we can compute the coordinates of a set of possible corresponding points in each of the other images. In general, these coordinates will not fall on integer pixel boundaries; thus some method of estimating the correct value of arbitrary points in the image is necessary.

The correct method for interpolation would be to convolve with a sinc function to remove higher order harmonics that are introduced in the sampling process. In practice the sinc function has a large support, which requires a large filter size and is therefore computationally intensive. A reasonable approximation is to use a Gaussian filter for interpolation. When combined with the LoG filter, this effectively produces an LoG filter with a larger standard deviation (the new σ is $\sqrt{\sigma_L^2 + \sigma_G^2}$), while interpolating the data as well.

In practice, for signals which have a cutoff frequency that is sufficiently less than the Nyquist limit (which we can ensure by choosing our LoG filter coefficients carefully), bilinear interpolation has proven to be sufficient for estimating actual image values at non-integer pixel locations. Bilinear interpolation also has the advantage of being easy to implement efficiently, since it only involves the four neighboring pixels. The gain in processing speed more than offsets the small loss in output quality.

7. Stereo Matching

The previous sections discussed how we can compute corresponding pixels between images that have high enough contrast to allow us to differentiate objects at different distances from the camera. What remains to be done is to search through the possible distances at each point and decide which one is best supported by the image data.

Ideally, we would just be able to compare the pixel values for each possible distance, and choose the ones that match best. If we assume that image noise is roughly Gaussian, then the best measure of the similarity of pixels is simply the squared difference between them. In practice, we find that outliers are actually much more likely than a Gaussian model would predict. One large factor that causes this to be the case for stereo matching is that the appearance of objects when viewed from different directions can be different. Two examples of when this occurs are specular reflections and occluding edges.

Since a good statistical model of such outlier points would be difficult if not impossible to construct, we are left with the problem of finding an error metric that is less sensitive to outlier points while being practical to compute. One such operator is the absolute value of the difference between pixel values.

Of course, since the pixel values are discrete integers between 0 and 255, the chances are good that several different pixels will match equally well even if we have the correct statistical model. With the addition of possible

image noise, it becomes likely that an incorrect disparity will match well. In order to compensate for this, we must make some further assumptions about the scene that we are viewing. The simplest assumption that we can make is that points in a small region of the image should all match in roughly the same way. This assumption is violated at occluding edges, and at points in the image with extreme slope compared to the reference plane.

The error metric for a particular pixel and disparity, for a single baseline (between cameras 0 and 1) is then:

$$E_{01}(x, y, d) = \sum_{i,j \in W(x,y)} |I_0(i,j) - I_1(i,j,d)| \quad (1)$$

where $I_1(i,j,d)$ is the appropriately interpolated value that matches $I_0(i,j)$ at distance d. $W(x,y)$ represents a window of pixels around the image point *(x,y)*.

One consideration is how to modify this metric for multiple baselines. The theoretically correct error metric assuming Gaussian noise would be to compute the variance of the set of image intensities in place of $|I_0(i,j) - I_1(i,j,d)|$. The metric corresponding to the absolute difference metric in the case of multiple baselines is:

$$\sum_{k=0}^{n} \left| nI_k(i,j,d) - \sum_{l=0}^{n} I_l(i,j,d) \right| \quad (2)$$

which is just the sum of the absolute differences from the mean (the variance would be the sum of the squared differences from the mean). Table 1 contains a list of different possible metrics and their computational cost. The quantity n is the number of cameras, and x is the cost of computing the maximum or minimum of two numbers. Since we are implementing this algorithm in modern computer hardware, multiply, add, and absolute value operations are assumed to be equivalent.

metric	formula	# of operations	with 3 cameras
absolute difference (variance)	$\sum_{k=0}^{n} \left\| nI_k(i,j,d) - \sum_{l=0}^{n} I_l(i,j,d) \right\|$	$5n-2$	13
		$\frac{3}{2}n^2 - \frac{1}{2}n - 1$	11
squared difference (variance)	$\sum_{k=0}^{n} I_k(i,j,d)^2 - \left(\sum_{k=0}^{n} I_k(i,j,d) \right)^2$	$3n$	9
absolute difference (reference)	$\sum_{k=1}^{n} \|I_0(i,j) - I_k(i,j,d)\|$	$3n-4$	5
squared difference (reference)	$\sum_{k=1}^{n} (I_0(i,j) - I_k(i,j,d))^2$	$3n-4$	5
all absolute differences	$\sum_{k=0}^{n} \left(\sum_{l \neq k} \|I_k(i,j,d) - I_l(i,j,d)\| \right)$	$\frac{3}{2}n^2 - \frac{3}{2}n - 1$	8
		3 camera special case	$4x + 1$

Table 1: Possible Error Metrics

Since whatever metric we choose will be evaluated for each pixel at each search distance, it is of critical importance that we choose a metric that can be evaluated with as few operations as possible in order to achieve an implementation that runs quickly.

As is shown in the table, the metric from equation (2), although theoretically correct, is one of the most computationally expensive. The use of variance as an error metric is motivated by the assumption that the pixels from each of the camera views are equivalent measurements of the same underlying property, and thus that their mean is the best estimate of that quantity and their variance is the best estimate of similarity.

In the stereo algorithm that we use for matching, one of the cameras (the reference camera, camera 0) is special. For each pixel of the image from that camera, we perform a search over a set of possible distances, comparing that pixel to different pixels from the other cameras. If we make the assumption that the pixel in the reference camera has the correct value (instead of assuming that the mean is the correct value), and we want to find the set of pixels in the other cameras that match it best, we get the metrics that are marked as (reference) in the table. These metrics, though not being strictly correct, are a good compromise that is about twice as fast and produces good results.

One other metric worth mentioning is the one marked "all absolute differences" in the table. This is the metric that results if the (reference) metric is expanded so that no particular camera is special. Though this metric has no mathematical basis, it can be computed very efficiently for the three camera case.

In general, we have used "absolute difference (reference)", though we have also experimented with "all absolute differences". The loss in accuracy caused was barely detectable, while the improvement in speed was large.

8. Obstacle Detection from Stereo Output

In order to detect obstacles, our system computes stereo results at each pixel for the "ground plane method" described in Section 3. We also compute the result for the traditional stereo method using vertical planes as illustrated in Figure 2. By comparing the matching errors for the best matches obtained with both methods, we can determine whether a given pixel is more likely to belong to a horizontal surface or a vertical surface.

The set of pixels that are determined to belong to ver-

Figure 9: Detected vertical surfaces

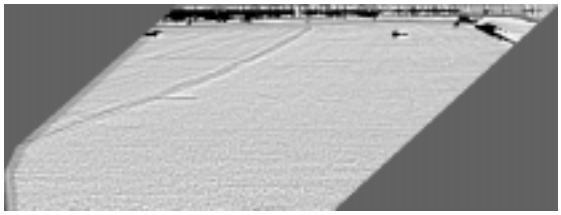

Figure 10: Detected Obstacles

tical surfaces is shown in Figure 9. This is the output for our example of Figure 4, which is an image of the road that we use for testing, with a 14 cm black obstacle placed at about 100 meters in front of the vehicle. The regions shown in the lower image are coded by the size of the difference between the minimum errors. Brighter regions indicate that the vertical match is much better than the ground plane match. Thus regions which appear white are most likely to be vertical, and black regions are most likely to be horizontal.

The obstacle, curb, and building in the background of the image are all identified as vertical surfaces by this method. Unfortunately, some regions of low image texture (such as the black patch in the center of the road) sometimes match well as vertical surfaces because of the lack of sufficient image texture for matching combined with image noise.

In order to remove such false obstacles from consideration, we use a very simple confidence measure. For regions which are actual vertical surfaces, we expect that the traditional stereo matching method will return a relatively large number of pixels at approximately the same depth. Conversely, if a region belongs to a horizontal plane, we would expect the traditional method to report a number of different depths. We use standard connected components labeling methods on the disparity image generated from traditional stereo matching, and count the number of pixels that belong to each region. In order for a pixel to be declared an obstacle, the size of its region must be above a certain threshold. By using this simple consistency check, we can remove most false positive detections.

Combining the images of Figure 7 and Figure 9 with this confidence measure, we get the detected obstacle output of Figure 10. Detected obstacles are shown in black. The 14 cm obstacle is detected, as well as the building in the background and the curbs on either side of the road. The curb in the background is too short and too far away

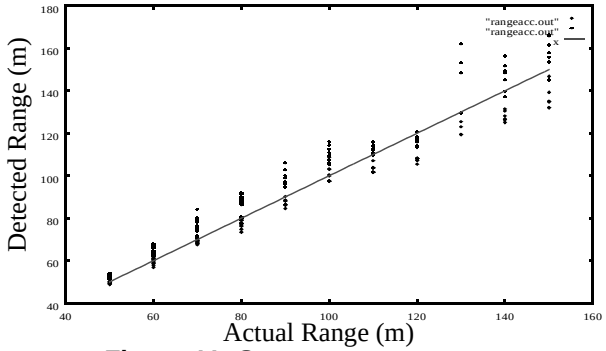

Figure 11: Stereo range accuracy

to be detected reliably.

9. Results

We have collected a set of test data using wooden obstacles of four different heights (9, 14, 19, and 29cm) and three different colors (white, black, and gray) at measured distances from 50 meters to 150 meters.

Figure 11 shows the accuracy of the detected range for all 12 obstacles. As expected, the measured range is very accurate when the object is close, and gets increasingly less accurate as the obstacle gets farther away.

The results of running the obstacle detection system are shown in Table 2. This table shows that we were successfully able to detect obstacles that are bigger than 9cm at up to 110m.

	Black 9cm	Black 14cm	Black 19cm	Black 30cm	Grey 9cm	Grey 14cm	Grey 19cm	Grey 30cm	White 9cm	White 14cm	White 19cm	White 30cm
50m	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
60m	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓
70m	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
80m	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
90m	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓
100m	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓
110m	✓	✓	✓	✓	×	✓	✓	✓	×	✓	✓	✓
120m	✓	×	✓	✓	✓	×	✓	✓	×	✓	✓	✓
130m	×	×	✓	×	×	✓	✓	✓	×	×	×	×
140m	✓	×	✓	✓	✓	✓	✓	✓	×	×	×	×
150m	✓	✓	✓	✓	✓	✓	✓	✓	×	×	×	×

Table 2: Obstacle Detection Results

Figure 12 shows an example trace of an obstacle detection run. The vehicle moved at a constant rate (about 25 km/h) toward a 14cm black obstacle of the type shown in Figure 9. The data was taken at 15 fps, and processed off-line. The obstacle is detected in every frame of the data, out to a maximum range of approximately 110m (which is the beginning of the data set).

Figure 13 shows the same type of trace, this time for a standard 12oz (350ml) white soda can. The soda can is first reliably detected at 57m.

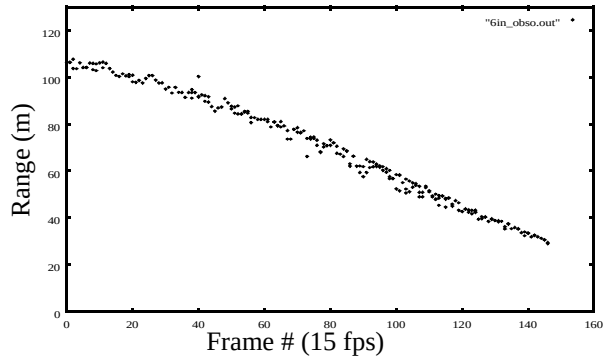

Figure 12: Detection trace for 14cm obstacle

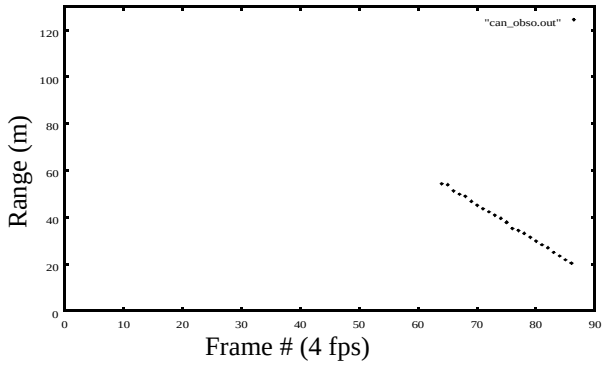

Figure 13: Detection trace for soda can

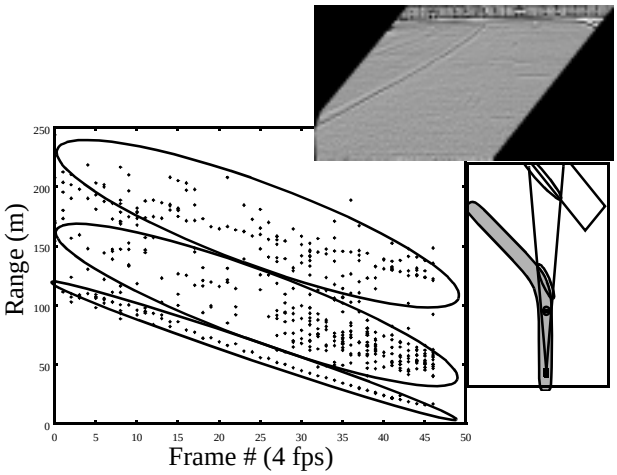

Figure 14: Other detected points

Each of the previous examples has shown only the detections that actually represented the obstacle. Of course, there are many more detected objects. A full trace is shown in Figure 14, along with an example image from the set and a diagram showing an overhead view of the scene. The detections can be divided into three sets, representing the obstacle, the curb behind the obstacle, and the building in the background. Also note that there are no false detections that are closer than the obstacle.

10. Conclusions

We have demonstrated an obstacle detection system that uses trinocular stereo to detect very small obstacles at long range on highways. The system makes use of the apparent shape of surfaces in the image in order to determine whether pixels belong to vertical or horizontal surfaces. A simple confidence measure is applied to reject false positives introduced by image noise. The system is capable of detecting objects as small as 14cm high at ranges well in excess of 100m.

11. Acknowledgment

This research was sponsored by the collaborative agreement between Toyota Motor Corporation and Carnegie Mellon University.

12. References

[1] P. Burt, L. Wixson, and G. Salgian. “Electronically Directed “Focal” Stereo,” *Proceedings of the Fifth International Conference on Computer Vision (ICCV ’95)*, Cambridge, Mass, June, 1995, pp. 94-101.

[2] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka. “A Stereo Machine for Video Rate Dense Depth Mapping and its New Applications,” *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR’96)*, June, 1996.

[3] Q.T. Luong, J. Weber, D. Koller and J. Malik, “An integrated stereo-based approach to automatic vehicle guidance,” *Fifth International Conference on Computer Vision (ICCV ’95)*, Cambridge, Mass, June 1995, pp. 52-57.

[4] L. Matthies and P. Grandjean. “Stereo Vision for Planetary Rovers: Stochastic Modeling to Near Real-Time Implementation,” *International Journal of Computer Vision*, 8:1, pp. 71-91, 1992.

[5] D. Marr and E. Hildreth. “Theory of Edge Detection,” *Proceedings Royal Society of London*, Series B, 1980, pp. 187-217.

[6] H. K. Nishihara. “PRISM, a Practical Real-time Imaging Stereo Matcher,” *MIT A.I. Technical Report Memo 780*, 1984.

[7] M. Okutomi and T. Kanade. “A Multiple-Baseline Stereo,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 4, April 1993, pp 353-363.

[8] L. Robert, M. Buffa, M. Hébert. “Weakly-Calibrated Stereo Perception for Rover Navigation,” *Proceedings of the International Conference on Computer Vision (ICCV ’95)*, 1995.

[9] T. Williamson and C. Thorpe. “A Specialized Multibaseline Stereo Technique for Obstacle Detection,” *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR ‘98)*, June, 1998.

[10] T. Williamson and C. Thorpe. “Detection of Small Obstacles at Long Range Using Multibaseline Stereo,” *Proceedings of the 1998 IEEE International Conference on Intelligent Vehicles*, Stuttgart, Germany, October 1998.

[11] T. Williamson. *A High-Performance Stereo Vision System for Obstacle Detection*. Carnegie Mellon Robotics Institute Ph.D. Thesis, CMU-RI-TR-98-24, 1998.