2008 IEEE Intelligent Vehicles Symposium
Eindhoven University of Technology
Eindhoven, The Netherlands, June 4-6, 2008

Lateral Vehicles Detection Using Monocular High Resolution Cameras on TerraMax™

Alberto Broggi, Andrea Cappalunga, Stefano Cattani and Paolo Zani
VisLab – Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma, ITALY
http://vislab.it
{broggi,kappa,cattani,zani}@vislab.it

***Abstract*—Autonomous driving in complex urban environments, including traffic merge, four-ways stop, overtaking, etc., requires a very wide range sensorial capabilities, both in angle and distance. This paper presents a vision system, designed to help merging into traffic on two-ways intersections, and able to provide a long detection distance (over 100m) for incoming vehicles. The system is made of two high resolution wide angle cameras, each one looking laterally (70 degrees) with respect of the moving direction, performing a specific background subtraction based technique, along with tracking and speed estimation. The system works when the vehicle is stopped at intersections, and is triggered by the high-level vehicle manager. The system has been developed and tested on the Oshkosh Team's vehicle TerraMax™ , one of the 11 robots admitted to the DARPA Urban Challenge 2007 Final Event.**

I. INTRODUCTION

After having seen five vehicles reaching the finish line in 2005, the Defense Advanced Research Project Agency (DARPA) moved its third-annual robot race Grand Challenge from the desert [**?**] into a city environment, calling it Urban Challenge [**?**]. The Urban Challenge features autonomous ground vehicles maneuvering in urban and sub-urban scenarios, where they had to execute merging into moving traffic, navigate traffic circles, negotiate busy intersections, avoid obstacles, follow lanes and handling parallel parking. Moving traffic was provided by several vehicles driven by professionals, as well as by the robots themselves, so robot-on-robot action was possible. Eighty-nine teams had applied to take part in the competition; DARPA accepted 35 of them, and only 11 were selected for the Final Event.

This paper presents an artificial vision system developed for the Oshkosh Team's vehicle TerraMax™ (Fig. 1). The Oshkosh Team is a partnership between academic and industrial members: Oshkosh Corporation, Teledyne Scientific Company, VisLab - University of Parma, Ibeo Automobile Sensor, and Auburn University. TerraMax was constructed from the chassis of the same MTVR truck used by Marine Corps, removing the posterior axle to improve maneuverability, and fitting it with drive-by-wire technology and a computer network that hosts the software applications necessary for autonomous navigation and sensing.

The Artificial Vision and Intelligent Systems Lab of the University of Parma developed the artificial vision systems that sensed the environment. The vision system is composed of 11 cameras, able to provide the following sensing information: lane/path detection, stop line detection, obstacle detection for straight driving and maneuvering, backward vehicle detection for possible lane changing, oncoming traffic detection when stopped at a stop line.

Fig. 1. The TerraMax™ vehicle

The system that performed oncoming traffic detection is discussed in this paper. It is called "lateral system" and it is composed by 2 high resolution wide angle cameras looking sideways. Some details, thresholds, and numeric parameters are kept confidential, as they are proprietary information.

The paper is organized as follows: section II presents the lateral system requirements, section III presents the hardware architecture of the lateral vision system, section IV contains a detailed description of the detection algorithm, section **??** discusses some experimental results and finally section **??** draws the conclusions.

II. LATERAL VISION SYSTEM CONSTRAINTS

When announcing the Urban Challenge, DARPA provided to the public the description of some typical situations the participants' vehicle were asked to deal with. One of the most challenging is the traffic merge, shown in Fig. 2.

Assuming the maximum speed of vehicles in an urban environment is $13m/s$ (30mph), it is possible to deduce the *minimum* detection range needed for traffic merge like this: $13m/s * 10s = 130m$. Such a long distance is far beyond the typical LIDARs detection range. Moreover, the direction of oncoming vehicles falls outside of the field of view of the cameras used by the other vision systems, that are focused on the vehicle front and back. Hence, a specific vision system was needed. The constraints we had to meet are:

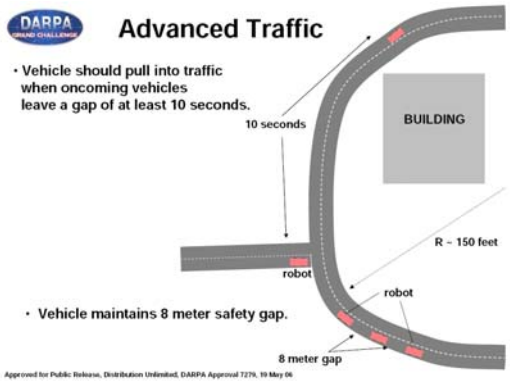


Fig. 2. Advanced traffic merge (Official DARPA document)

- long distance detection capabilities, intersections not always perpendicular;
- detection of *moving* oncoming objects, with speed estimation;
- real-time requirements must be met: 10Hz is the minimum processing rate.

The first constraints led us to choose high resolution and wide field of view cameras, in order to cover the several shapes an intersection can have, furthermore the cameras are mounted in a convenient position, as described in Sec. III. Since this system is used at intersections when the vehicle is standing, the detection of moving objects is made via a *background subtraction* based detection, as described in Sec. IV, along with tracking and speed estimation.

III. HARDWARE SETUP

The cameras are equipped with a 1920x1080 pixels RGB sensor. As mentioned in Sec. II, the wider field of view, the better. Unfortunately, short focal length lenses for 1" sensors are too distorting for our purposes. After some tests we finally mounted 8mm Kowa lenses, that provide a good trade off between wide field of view ($85°$) and low distortion.

The cameras are mounted on a bar just in front of the vehicle, at a 165cm height from the ground, pointing $70°$ away from the truck longitudinal axis (Fig. 3).

If an object w meters wide at a distance of d meters is arriving at the intersection where we are standing, we are able to estimate its size in pixel sp on the image as in the following:

$$sp = (focal\_length * w)/(d * pixel\_size) \qquad (1)$$

so a generic vehicle ($w \simeq 1.5$m), 130m far from the truck, will be about 13 pixels wide on the image. Consequently the size of the smallest moving object the algorithm must be able to detect was set to 10 pixels.

Fig. 3. The two lateral cameras

IV. ALGORITHM

A. Architecture

To deal with the large image size provided by the high resolution camera, without overrunning the strict time specifications described in Section I, we designed a *hybrid* multiresolution processing method. Two separate processings are applied on each captured image: the first one processes an horizontal slice of the full resolution image, while the second one operates on a downsampled version of the whole image. Each process is executed independently of the other one, and the results are collected and fused together. In this way the computing weight is considerably reduced, if compared with a single full image processing, and at the same time we can still take advantage of the high resolution on far away zones, without precluding to check also near areas.

The multi resolution process is triggered by a specific network message, sent by the high-level vehicle manager: images are processed when the vehicle is in "stopped" state, where the speed is guaranteed to be 0. While, if the system do not receive any vehicle manager message nor inertial sensor data, it will keep computing in low resolution only; in this way the system is able to provide a minimum level of detection, at least at short distances.

Since the processings described above are totally independent of each other, in order to reduce the execution time we exploited the dual core CPU, by executing the two processings in separate threads. This structure is thus replicated on each side, right and left, and both are put in execution again on separate threads; the final high level system structure diagram is represented in Fig. 4.

It is important to notice that all the process is done on gray images, ignoring the color information to keep the computing time low.

B. Background Subtraction

The basic layer of the system is a *background subtraction* algorithm [**?**] [**?**], implemented with the additional features explained in this paper, in order to overcome some problems encountered during the development; for example even if the algorithm is executed only when the truck is stopped, the engine vibrations are so strong to make the camera oscillate (estimated in more than ± 5 pixels vertically), causing non-zero differences when subtracting background objects in a classical . Moreover, in a complex urban scenario many

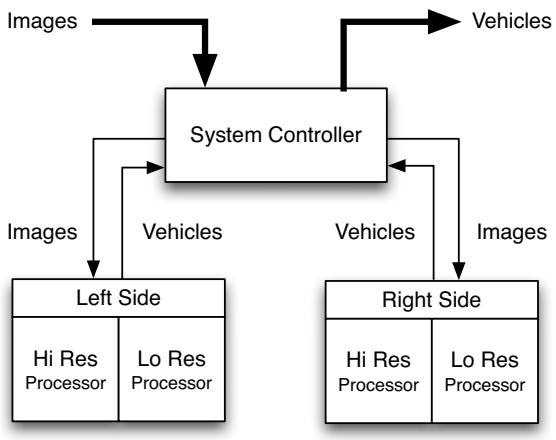


Fig. 4. System structure diagram

objects can change their appearance periodically, like flags, trees and also traffic lights. Both vibrations and periodic movements can cause false positives detection. While the second problem can be solved in the tracking step (see Section IV-E) by filtering out objects that do not change their average position in time, the first one is more subtle. Due to vertical oscillations, the non-zero responses in the difference image often appear like wide horizontal edges, hard to uniquely be located in world coordinates: this prevents the tracker from filtering them out on the basis of a persistent position in time.

For these reasons, the vibration problem needed an ad-hoc solution: the first step of the algorithm is a per-pixel difference between the last acquired frame and a reference image, built as the time weighted *average* of the previous N frames.

The weights must be computed in order to assign higher importance to recent frames against far ones; furthermore they have to preserve the global image brightness. We can then build the desired weights array, with a linear decreasing curve, by computing a k factor satisfying the following equation:

$$k \cdot \sum_{i=0}^{N-1} (N-i) = 1 \tag{2}$$

where N is the number of frames considered during the operation; so, according to equation (2) the resulting weights are all the multiples of the constant k computed in this way:

$$k = \frac{2}{N(N+1)}. \tag{3}$$

Using these weights, the j-th pixel of the reference image is computed as:

$$I_{ref}[j] = \sum_{i=0}^{N-1} k \cdot (N-i) \cdot F_{t-i}[j] \tag{4}$$

where F_{t-i} are the previous acquired frames.

By applying these operations, the image we obtain is a pseudo-background that can be used to reveal movements, with the important features of noise-reduction and high vibrations attenuation; we have though to notice that N (the number considered frames) should not be too high, to avoid too strong blurring effects, that may lead to trace moving objects impressed on the reference image for too long, and to a deterioration of the detection performance.

Now it is possible to compute the absolute difference pixel-by-pixel, and immediately after, to apply a static threshold to quantize the resulting image in two levels:

$$I_{diff} = \begin{cases} 255 & \text{if } |I_t - I_{ref}| \geq threshold \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

In Fig. 5(c) the final result of this stage is shown.

C. Detection

Still focusing on the aim of keeping the vehicle detection process simple and computationally light, due to time constraints, all zones revealed by the difference are not bounded by a labeling algorithm; they are instead enclosed into bounding boxes with a simpler histogram approach, based on the assumption that there could not be vehicles stacked vertically (one upon another) on the image.

moving vehicles. Hence, to reveal all the zones in which there has been some movement, we make a vertical histogram of the binarized difference image, counting for each column the number of white pixels; then we create several slices of the image, one for each zone where the histogram is higher than a threshold (Fig. 5(d)).

Once we have obtained the vertical slices, the upper and the lower bounds of the moving object are searched inside those, by computing a new histogram: this time it will be horizontal, looking for the first and the last row where the histogram is over the threshold. Finally the horizontal coordinates of the slices, and their upper and lower bounds, are used to create a bounding box for each moving object (Fig. 5(e)).

A further operation we apply to the created bounding boxes is a shadow eraser filter: this because also the pixels corresponding to the shadows projected by vehicles on the ground are, at this stage, seen as moving objects. In low sun conditions these shadows may cause an error in position estimation of more than 1 meter. Since the obstacles provided by the lateral system have to be fused together with other sensors' output, it is important to be as much accurate as possible, to do not introduce noise into the higher-level fusion step. To remove shadows we suppose that the height of a lateral projected shadow of a vehicle is smaller than the vehicle itself; hence the shadow eraser filter analyzes the slice of the histogram relative to each bounding box, and determine if there are zones, at both sides of the slice, which have an histogram height heavily lower than the box's height; in that case the filter will cut that part, as shown in Fig. 5(f)(g) which

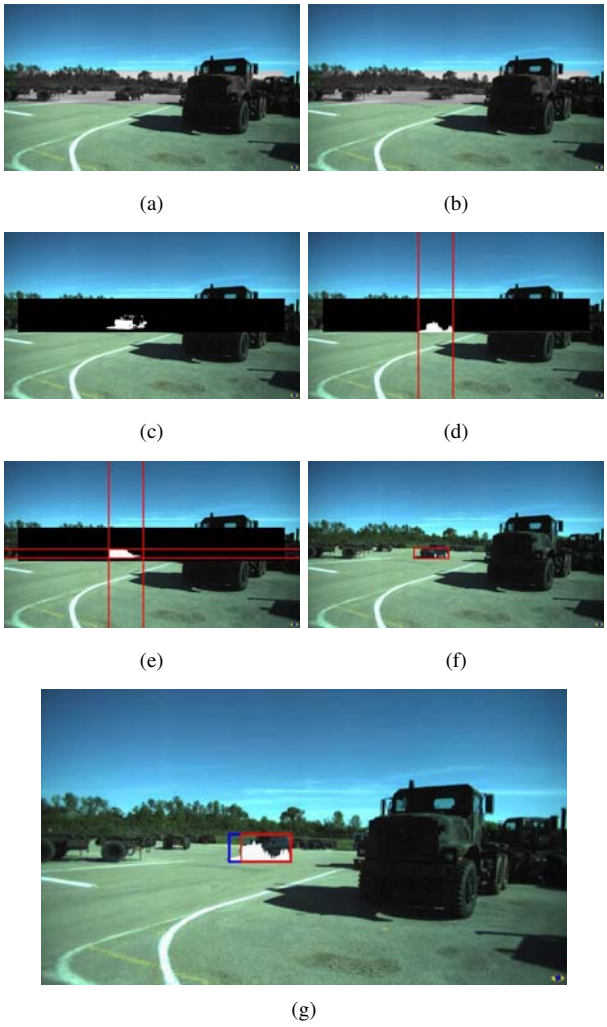


Fig. 5. Preliminary background subtraction and detection: (a) the last captured image, (b) the reference image, (c) the binarized difference, (d) the vertical histogram, (e) the horizontal histogram made on the slice computed on the basis of the vertical histogram, (f) the bounding box including the vehicle's shadow and (g) the final detection result.

presents how moving objects are shown at this preliminary stage.

D. Progressive Background Generation

We have seen how motion detection is based on a background subtraction algorithm, but we have also shown how that step depends on the difference between two images; the difference operation is able to enhance pixels where the image has changed, but only on the edges of the moving object.

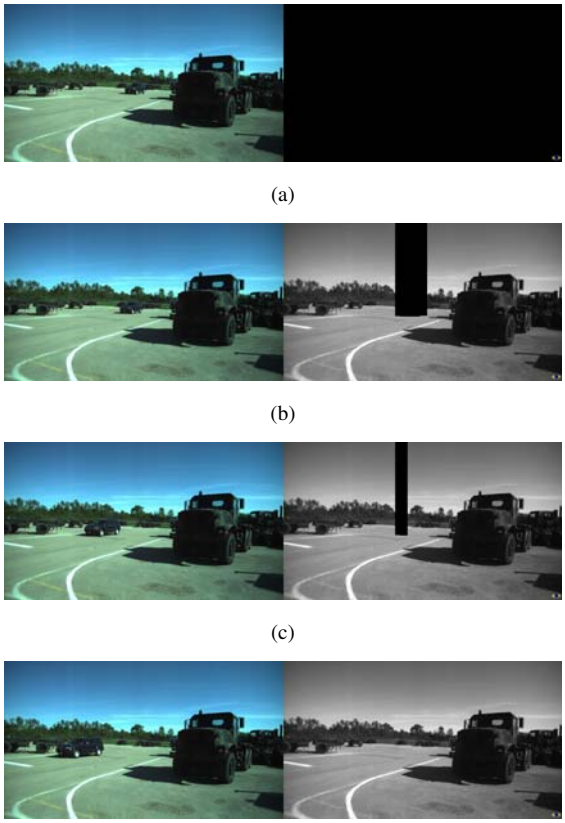
The proposed solution tries to replace the reference image, as computed in equation (Eq. 4), with the background of the scene as it would appear if no moving objects were present. This can be easily achieved when working in a-priori known environments, like in traffic monitoring or in laboratory testing, just taking manually the best background image, and using it indefinitely. Unfortunately in our case this is not possible, since the cameras are installed on a moving vehicle, hence we have studied a method to *progressively* generate the background image every time the vehicle stops at an intersection, working just on the acquired stream. The basis of the background generation stands on the assumption that vehicles in the scene are in motion, so they will not hide the same part of the scene's background forever.

The starting point of the algorithm is the list of bounding boxes created by the detection phase and an empty background image (Fig. 6 (a)). At each execution of the process, one per acquired frame, the black zones of the background image are updated in this way: if their corresponding pixels are now not hidden by some moving object (and this information is easy to retrieve by using the list of bounding boxes), they are filled with the pixels of the input image.

It is clear how this process is iterative and incremental (Fig. 6 (b),(c),(d)) and takes a couple of frames to complete, depending on the speed of vehicles. However, it is possible to take advantage of having also a partial background, by using it immediately with just some adjustments: it is possible to substitute the reference image with the background image during the background generation process. This approach is very efficient, but still keeps all noise and vibrations which affect the input image; hence we decided to use again the reference image, instead of the input one, for the background image filling. In this way we can take advantage of the generated background image and, at the same time, of the vibration attenuation effect given by the reference image, obtaining a better defined obstacles' shape. Finally, once the background is completed, it can be considered frozen and no more filling are needed. There is hence a problem when a vehicle that was initially static starts moving; this situation can create a standing "ghost" on the system output. In fact, once the started vehicle has completely detached from its initial position, the binarized image will presents two blobs: one corresponding to the real moving vehicle and one corresponding to the shape of the same vehicle, in the background image. In this case the higher level tracking algorithm will discard the standing ghost, since it is not moving at all, and its image area will be replaced by the corresponding area in the reference image. Another possible problem that affect this background based approach could be given by fast illumination changes, that can results into several big blobs into the difference image. 1 these blobs has not a motion compatible with a vehicle, thus these effects can be immediately smoothed by the tracker, which can also invoke a background refresh. Moreover the vision system is equipped with a software automatic gain and exposure control that strongly attenuates illumination related issues.

E. Tracking

In order to stabilize the algorithm output, a tracking stage is required to reduce false positives/negatives and, at the same time, to estimate vehicles' speed by exploiting their history. The simplicity guideline has affected also this part of the algorithm, leading up to develop a simple tracking system which working on the positions of the detected bounding



(a) (b) (c) (d)

Fig. 6. The progressive background generation process. In (a) the background is empty. In (b)(c) how the background is generated as moving objects change their position. In (d) the final background.

boxes, instead of using other heavy feature-based systems, well known in literature [**?**]. The developed tracking system operates in two steps: the motion analysis phase, to search for a mapping between the list of previously detected (and tracked) vehicles and the list of the new bounding boxes, and a filtering phase, whose goal is to discard all objects with a motion reckoned incompatible to a vehicle's one.

1) Motion Analysis: As introduced above, with motion analysis we want to find the relations between the existent list of tracked obstacles and the current bounding boxes, finding for each previously known vehicle its new position. The approach we adopted to create these relations is based on the assumption that the bounding boxes belonging to a single object, in two adjacent frames, must have an overlapped area. The case of two consecutive disjoint bounding boxes corresponding to the same oncoming vehicles is rare, due to the perspective mapping. Furthermore we assume that the most relevant point of the bounding box belonging to a vehicle is the lower left corner for the right camera images, or the lower right for the left camera (see Fig. 5(g)). We chose those points because they probably correspond to the closest part of an approaching vehicle and, at the same time, they are the most accurate points we can consider belonging to the vehicle, due to the monocular vision limits. Hereinafter with "linchpin corner" we will refer to that points.

Under these assumptions we can assert that, if a bounding box of the previous list has an overlapped area with one, and only one, of the current boxes, they can be associated; if the considered box is overlapped with several current boxes, we can choose the one with the nearest linchpin corner as the most appropriate. When the associations are created, each obstacle tracking information is updated with its new position and its last motion vector, calculated between the current linchpin corner and its previous one. The other cases, like vehicles arrival/departure or intermittent detections, are handled in part with an hysteresis window put on the acceptance/discard phase, like any other tracking algorithm, and in part by the filtering phase of the tracker (see Section IV-E2).

2) Filtering: The filtering phase of the tracker works on the statistics data of each obstacle, calculated on a maximum time window of 20 frames, in particular on its motion vector's average length and its direction's variance; these two parameters are very important in order to determine if the motion of a tracked object is compatible with a vehicle's one. In fact, by analyzing the average length of the motion vector, it is possible to figure out if a detected object is moving or not: if the average length is equal to 0, we are in the case of still object and we can assume that the obstacle may be a false positive, like the "ghost" obstacles discussed in paragraph IV-D, and it can be discarded as well.

The variance of the motion vector's direction is, instead, used to infer some information about object's motion regularity; in fact a small value for that index will correspond to a uniform movement, the one we expect from a vehicle which is moving upon a street, while a greater variance will correspond to a quite random motion, maybe belonging to some swinging vegetation or similar. So all these considerations about motion statistics are represented by a score assigned to each object, that is increased/decrease according to the statistic indexes analysis; finally the score is used to update the status of an obstacle, that can be one of the following: *approved*, *keeping*, and *discarded*.

F. Coordinates conversion

The part of the algorithm explained so far shows the whole process we used to identify vehicles on images, but a further step is needed to provide the real world position of the detected vehicles to the higher level step. Hence the pixel coordinates of each obstacle must be converted into TerraMax^{TM}reference frame, set on the center of the front bumper projected on the ground. The approach we used to this conversion, is an Inverse Perspective Mapping (IPM) [**?**]: assuming to know all the camera's parameters, the IPM allows to convert coordinates from image to world by applying a geometrical transformation.

Once we have assigned the position expressed in meters to each detected moving vehicle, the final step will be to send them to the appropriate higher level service, so they can be used by the navigation system.

Moreover, the world position of the vehicles is also used

to estimate their speed, by considering the ratio between the traveled distance and the time elapsed.

However the coordinate conversion stage is very critical, because of the high noise sensitivity of the IPM, especially at great distances (like 100m). In fact an error of a single pixel, during the calibration process, can be reflected on an error of several meters in the obstacles' position determining.

V. RESULTS

The algorithm showed good detection rate during test sessions, especially with respect to far vehicles; however some false positives/negatives are present, even if the introduction of the tracking system has drastically reduced their number. As far as time is concerned, the overall system takes less than 100ms on an Intel Core Duo 2.0GHz (T2500) to process both right and left images in multi resolution mode. Some results are shown in Fig. **??**.

VI. CONCLUSIONS AND FUTURE WORKS

This paper presented a moving vehicle/object detector, using monocular high-resolution cameras and based on an enhanced background subtraction technique. The system was developed and tested on the Oshkosh Team's vehicle TerraMax^{TM} and was specifically designed to help it during traffic merge manoeuvres at the DARPA Urban Challenge 2007. The main difficulty of this task was the detection of long distance oncoming traffic, due to the pixel size a vehicle has at such distances and to the high vibrations affecting the cameras.

The lateral system demonstrated great potential during the TerraMax^{TM} development and testing: it was able to accurately detect moving objects further that 100m, to estimate their speed and met the real-time constraints. However some improvements can be applied to this system, in order to overcome basically two issues: the high sensitivity to camera calibration parameters during the vehicles' position computation, and the fact that currently the system does not discriminate between vehicles, pedestrians, and others. The first one can be mitigated by exploiting inertial information provided by the INS system, and apply an on-line correction to calibration parameters; while the latter could be solved by introducing an object classification system to notify which objects have to be considered and which have to be discarded.

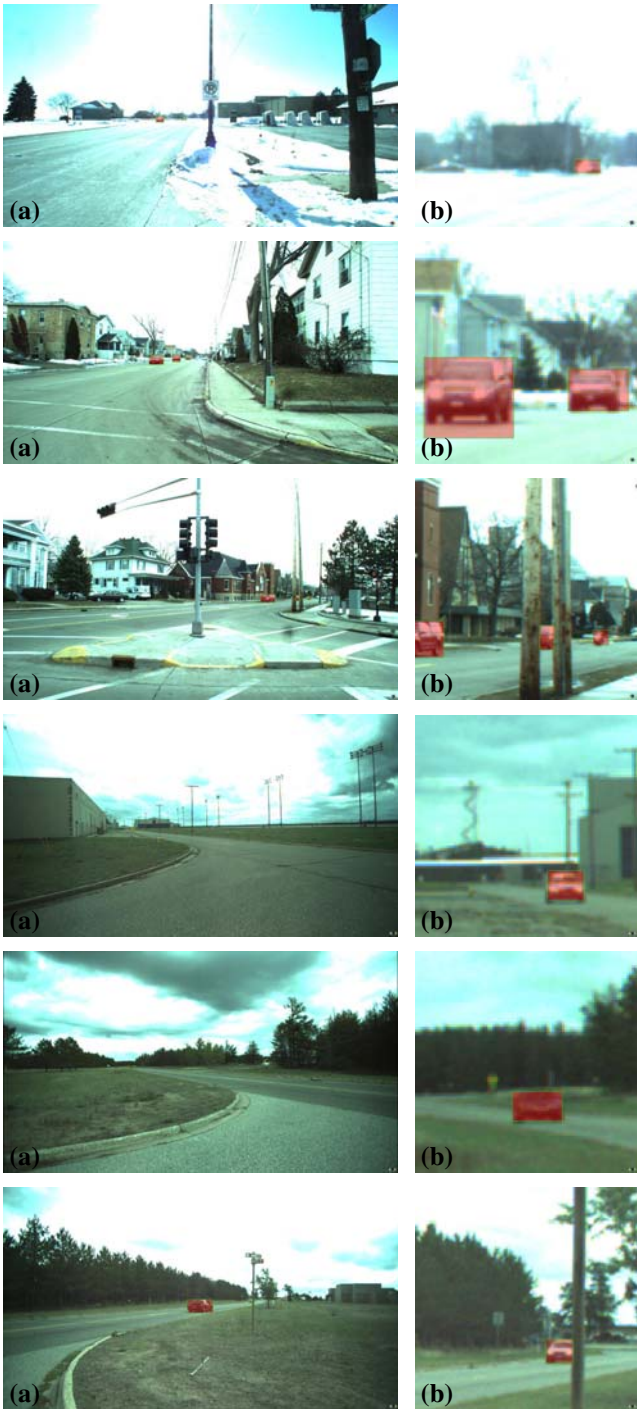


Fig. 7. Detection results. For readers convenience, the left side shows the complete result images and the right side the same images zoomed in to highlight far away detected objects.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the contribution of Oshkosh Corporation, Teledyne Scientific Company and all the members of Oshkosh Team.

REFERENCES

[1] C. Caraffi and S. Cattani, "VisLab at the Grand Challenge," *IEEE Computer*, vol. 39, no. 12, pp. 36–37, Dec. 2006.

[2] Defence Advanced Research Projects Agency (DARPA), "Urban Challenge 2007," Avaiable http://www.darpa.mil/grandchallenge.

[3] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 10, pp. 1337–1342, 2003.

[4] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, vol. 02, pp. 302–309, 2004.

[5] C. Tomasi and T. Kanade, "Detection and tracking of point features," Tech. Rep. CMU-CS-91-132, Carnegie Mellon University, 1991.

[6] Massimo Bertozzi, Alberto Broggi, and Alessandra Fascioli, "Stereo Inverse Perspective Mapping: Theory and Applications," *Image and Vision Computing Journal*, vol. 8, no. 16, pp. 585–590, 1998.