Space variant vision and pipelined architecture for time to impact computation*

Fernando Pardo Isaac Llorens Francisco Micó José A. Boluda
Departamento de Informática - Universitat de València
Fernando.Pardo@uv.es, Issac.Llorens@uv.es, Francisco.M.Mico@uv.es, Jose.A.Boluda@uv.es.

Abstract

Image analysis is one of the most interesting ways for a mobile vehicle to understand its environment [1]. One of the tasks of an autonomous vehicle is to get accurate information of what it has in front, to avoid collision or find a way to a target. This task requires real-time restrictions depending on the vehicle speed and external object movement. The use of normal cameras, with homogeneous (squared) pixel distribution, for real-time image processing, usually requires high performance computing and high image rates.

A different approach makes use of a CMOS space-variant camera that yields a high frame rate with low data bandwidth. The camera also performs the log-polar transform, simplifying some image processing algorithms. One of this simplified algorithms is the time to impact computation. The calculation of the time to impact uses a differential algorithm. A pipelined architecture specially suited for differential image processing algorithms has been also developed using programmable FPGAs.

1 Introduction

The aim of this research work is to analyse the movement of objects in front of a self-moving vehicle using image processing techniques. The problem of a car in a road has been studied to find the real-time constraints for this specific problem. The combination of space-variant imaging, specialised pipelined image processing architecture, and the implementation of a time to impact algorithm, has shown to be good enough to meet the real-time requirements of the aimed problem.

As images taken by any camera are two-dimensional objects, it is difficult to extract information about the depth and distance of the objects in front of the vehicle. If the vehicle is moving, a dynamic analysis of the changing environment can give information about the distance from objects. This dynamic computation is also useful when objects in front of the vehicle are also moving. Dynamic calculation usually requires real-time constraints that will depend on the vehicle speed and object movements in front of it.

Retinal sensors widely expand the possibilities of real time applications [6]. Advantages refer to the special polar structure of the resulting images that allows simplifying algorithms based on polar characteristics of the images.

Cameras provided of such sensors can be used to evaluate motion due to their small size and portability, much more compact than conventional cameras. They generate images with a selective reduction of information, more concentrated near its optical center, the fovea, and less accurate towards the periphery. The reduction of redundant information diminishes the computational burden of most algorithms. Thus, retinal sensors become an interesting device for robotics and real time systems for navigation.

One possibility for a space-variant camera is the log-polar mapping. The focal plane of this camera has two areas with analogous names as the human eye: the retina is the outer part and it occupies most of the sensor area, in this part the distance of pixels to the sensor center increases exponentially thus decreasing pixel resolution toward the periphery; the fovea is the small central part with the highest resolution, it follows the same polar pixel distribution though pixel distance to the centre increases linearly instead of exponentially. This log-polar transformation can be shown in figure 4 where the circle represents the focal or retinal plane, while the cartesian represents the computation or cortical plane. The computation plane is the transformation of the focal plane to the computer memory; this computation plane is what the computers *sees* of the environment.

The use of this kind of log-polar images reduces the amount of data to be processed thus allowing higher image processing rates. In order to have even higher image rate, a pipelined architecture is proposed. This architecture is specially suited for image processing algorithms requiring spatial and/or time derivatives. Each element in the pipeline has local memory to store last images thus allowing time derivatives calculation.

*This research work has been supported by the Generalitat Valenciana project ref. GV99-116-1-14

2 Image processing and acquisition architecture

The overall architecture is shown in figure 1. The elements of the system are: the log-polar camera, the specialized pipelined architecture, and a common host computer (PC). Log-polar images taken by the log-polar camera enter the host computer that bypasses the images to the pipeline of processing elements (three for the time to impact experiment). The processor pipeline processes the image flow providing the results to the host computer, where other higher-level tasks are performed (image segmentation). A description of each element follows.

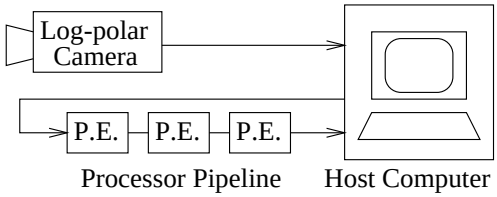

Figure 1. Overall system architecture.

2.1 Log-polar camera

Figure 2 shows the main blocks of the log-polar camera. The most important elements of the camera are the *control unit* and the *log-polar sensor*.

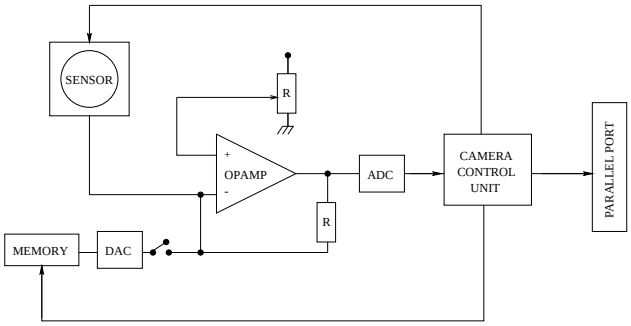

Figure 2. Log-polar camera.

The control unit is an Altera CPLD of the FLEX 8000 family. It is a low-cost programmable logic device with 4000 usable gates. The main function of the control unit is to communicate the sensor with the PC parallel port. This unit also generate the addresses for the log-polar sensor to acquire images at specified and programmable rates.

The log-polar sensor has been developed using CMOS technology [2]. In the retina (outer area), each ring has 128 pixels whose width grows toward the periphery. The fovea, unlike the retina, does not have a fixed number of pixels by circumference; the amount of pixels diminishes toward the centre up to only one pixel. The CMOS sensor is mounted on a 68-pin PLCC socket.

CMOS technology allows random access to any pixel of the sensor. The control unit indicates to the sensor the pixel to be read. An operational amplifier is necessary to amplify the small signal from the sensor and to subtract the Fixed Patter Noise (FPN). This OPAMP also changes the gain and adjust the signal to the range of the A/D converter. The converter is a CMOS 10-bit 20 MSPS A/D converter. Though it is a 10-bit converter, the camera only makes use of the 8 most significant bits for precision. The A/D converter gives the digital value of the pixel to the control unit.

The host computer has a special interface to communicate with the log-polar camera and the pipeline of processors. It is a custom PCI card under development at this moment [3]. Instead of this card we use the Enhanced Parallel Port (EPP) mode of the PC parallel port to connect the camera and the processors through a parallel cable to the PC. The EPP mode has a typical transfer rate from 500 Kbyte/s to 2 Mbyte/s. This performance is achieved thanks to the new registers added to the EPP mode that allow 32 bit transfers with a single instruction. EPP allows communicating both data and addresses between the camera and the PC using the EPP Data Register and the EPP Address Register respectively. The camera sends data to the PC, and the PC can also request information (pixel, new image, etc.) to the camera sending addresses.

Log-polar images have a resolution of 128 pixels per ring and 76 rings; this is a total of 9 KBytes per image. Using the EPP at 1.5 Mbytes/s (half of its maximum performance) it is possible to transfer up to 150 images/s, more than enough for our application.

The camera makes use of a memory and a D/A converter for the automatic circuitry for offset compensation (FPN cancellation circuit). The memory stores the FPN image. The D/A converts this image to analog values that are subtracted to the image from the sensor in the OPAM. The D/A converter is a standard 8-bit monolithic digital-to-analog converter. This subtraction cannot be performed in the digital domain, since accuracy is lost. To digitally make this cancellation it is necessary to acquire images with resolutions higher than 8 bits to prevent accuracy losses; also the FPN image has to be stored using higher precision.

2.2 Pipeline of processors

The inclusion of a specific architecture in the processing flow of the system reduces the tasks to be performed by the central computer. A pipelined architecture has been chosen and implemented since it better suits image-processing algorithms where several simple operations take place on the same image flow [4]. Dynamic analysis is also a con-

cern in the kind of applications we aim; this is the reason to put local memory in the processing elements. Two dual port memories are employed to communicate one processing element to the next, storing at the same time intermediate results. Figure 3 shows the processor pipeline (only two processing elements drawn) with the structure of each processing element.

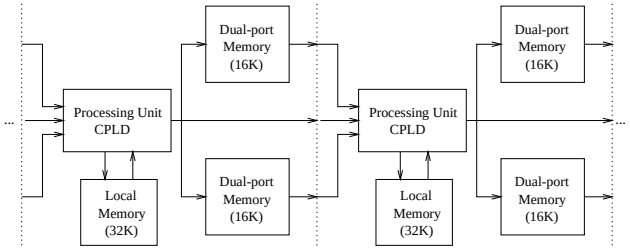

Figure 3. Pipeline of two processors.

The processing unit has been designed [5] with an Altera FLEX8000 family CPLD that has similar complexity and capacity as common FPGAs. The use of a CPLD allows the hardware implementation of image processing algorithms obtaining higher processing speeds. It is also possible to reconfigure the algorithms on line, since the CPLDs of this architecture can be programmed in few milliseconds by the host computer. Several tasks acting at different times can be deferred to the processor pipeline to relief the central processor of heavy work.

The algorithm for the time to impact computation has been successfully implemented in this pipeline of processors. It has been necessary to use three stages of processing elements: the first one makes a general smoothing of the input image, the second one calculates the spatial and temporal derivatives, the last one calculates the time to impact dividing the spatial and temporal derivatives.

The three stages work in pipeline synchronized with a 16.6 MHz clock signal. The smoothing stage takes 4 cycles to complete; the temporal derivative and spatial gradient stage takes 6 cycles; finally the division stage takes 16 cycles. This last time fixes the total computation time per pixel in the pipeline. It means that processing one image takes around 12 μs to complete, thus up to 80 images per second can be processed.

3 Time to impact computation

An approaching object following the optical axis of the retinal sensor will experiment an apparent scaling and its progress will be viewed as an expansion motion. Consequently, its movement can be represented by a radial optical flow from the center of the sensor towards its edges. Calculations of such a movement are simplified if it is characterized by means of a log-polar representation.

$$\begin{cases} \xi = \log r \\ \gamma = \theta \end{cases} \tag{1}$$

Scaling in this system become a simple translation as it can be seen in figure 4, where a ring approaching to the camera is shown.

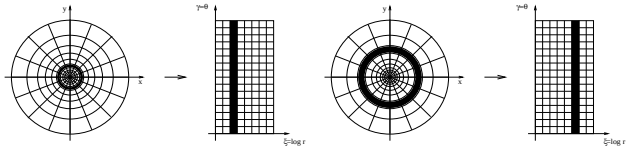

Figure 4. Approaching object in a log-polar mapping.

Therefore, it is possible to establish a relationship between the speed of an approaching object and its apparent radial speed. Let us suppose a P point that is coming to the objective of the sensor in a direction parallel to its optical axis, with speed $W(t)$ as shown in figure 5.

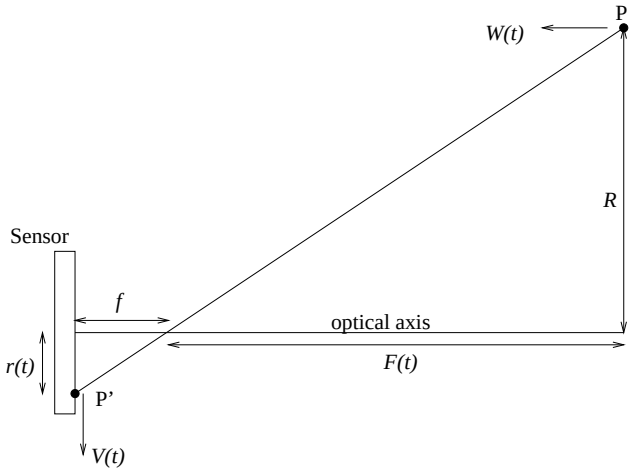

Figure 5. Time to impact calculation.

Being f the objective focal, R the distance from P to the optical axis (and therefore constant), $F(t)$ the distance from the projection point of P on the optical axis to the objective focus, and lastly, $r(t)$ the distance from the optical axis to P', the image of P on the sensor, then the following simple relationship is accomplished:

$$\frac{f}{r(t)} = \frac{F(t)}{R} \tag{2}$$

So, $V(t)$, the apparent speed of P', will be related to $W(t)$ by:

$$V(t)\frac{f}{r^2(t)} = \frac{W(t)}{R} \tag{3}$$

Combining these two equations, it is possible to calculate the *time to impact* τ; it is to say, the time the object will invest to collide with the focus of the system:

$$\tau = \frac{F(t)}{W(t)} = \frac{r(t)}{V(t)} \tag{4}$$

Thus, the time to impact can be expressed as the relation between the distance of P' to the center of the sensor and the radial component of the speed in the image plane. As a consequence, it is possible to trace an impact time map, simply dividing the ratio of each image point between the optical flow in that point. When these equations are translated to the log-polar plane, equation (4) becomes even simpler since the term $r(t)$ disappears (cancelled by the coordinate change in $V(t)$).

The $V(t)$ is the *optical flow* and there are several methods for its computation [7]. In Cartesian coordinates it becomes a difficult task since the speed $V(t)$ has two components V_x and V_y. In log-polar coordinates it also has two components (V_ξ,V_γ), but as far as we are just focusing the problem of approaching objects, the polar optical flow component V_γ is always cero. Having just one component to calculate simplifies the problem since the Horn equation [8] can be directly employed to calculate the optical flow and its inverse, that gives the time to impact as shown in the following equation:

$$\tau = -\frac{1}{B}\frac{\frac{\partial I}{\partial \xi}}{\frac{\partial I}{\partial t}} \tag{5}$$

where B is a constant that depends on the sensor geometry, I is the image intensities (image), $\frac{\partial I}{\partial \xi}$ is the image derivative along the ξ axis (spatial derivative) and $\frac{\partial I}{\partial t}$ is the image derivative with respect to the time (time derivative). These two derivatives are the only calculations to be made on the image, and they are as simple as subtractions. The use of log-polar coordinates and its application to the special problem treated allow this great simplification respect to the same problem solved in Cartesian coordinates.

4 Experiments

A simple experiment has been performed using laboratory conditions. In this experiment we move two mouse balls toward the camera nearly following its optical axis. The first, midle and last log-polar images of the experiment are shown in figure 6.

The black ball started its movement at 41 cm away from the camera at a speed of 2.54 cm/frame. The white ball started at similar position (43 cm) but had slower speed (2 cm/frame). A total of 16 frames were taken for the experiment. The result is shown in figure 7 where the experimental points along with the theoretical lines are shown. These results have been obtained after a small higher level post-processing that included image segmentation to separate both balls, wrong points filtering (points with low time derivative) and final mean calculation among points.

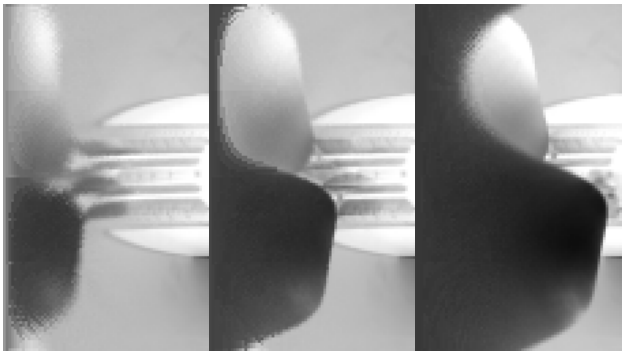

Figure 6. First, middle and last frames of the sequence.

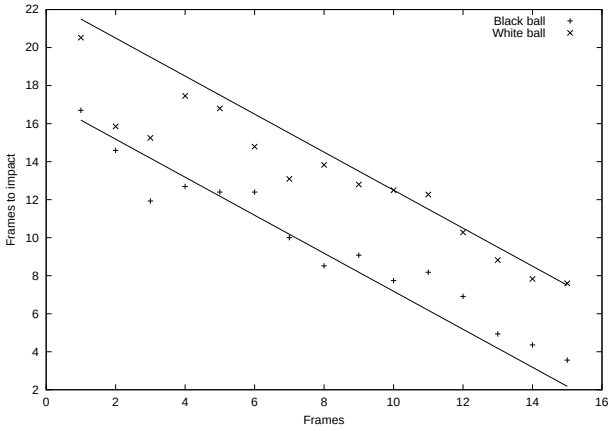

Figure 7. Time to impact for the two balls.

Despite the small differences between the speed of the balls it is possible to see this difference in figure 7. When the balls are away from the camera (first frames) the time to impact calculation is very spread since the error at long distances is high. This error become smaller when the balls approach the camera.

In order to test this same algorithm in road environments, a theoretical experiment has been studied. Suppose various items (four circular objects of different dimensions and speeds) moving towards a polar camera system, following its optical axis:

Object	A	B	C	D
Radius (m)	1.6	0.8	1.6	0.8
Speed (km/h)	50	50	100	100

Initially, all these objects are placed at barely a hundred meters from the focal of the system. The estimated times to impact (in seconds) of these objects are shown in figure 8, next to their ideal times.

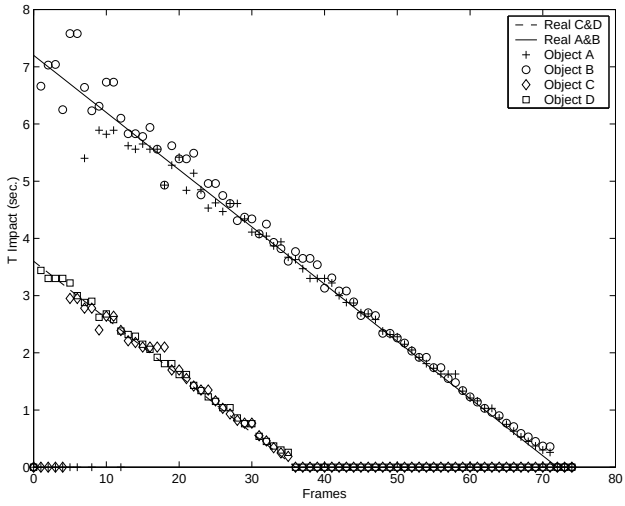

Figure 8. Time to impact of four approaching objects.

This experiment shows that just with few images it is possible to start calculating the time to impact. It must be noticed that camera is set to take images at a rate of 10 frames per second, though it can be increased up to 80 frames per second at wish, on account of the hardware capabilities of the pipeline of processors used in this experiment. In this experiment, though theoretical, it is possible to see the higher dispersion of the time to impact calculated at long distances. This is the same effect shown in the other experiment and it is due to the finite resolution of the camera.

5 Conclusions

The fast motion of vehicles in roads is a high demanding computation problem, which usually requires severe real-time constraints. Detecting time to impact of an approaching car, in time to avoid collision can be of interest to increase current car safety.

In order to solve the problem of calculating the time to impact of objects in road like environments, we put together several strategies to minimize the time of calculation. First we employ a log-polar camera that selectively reduces the amount of data to be processed without loosing accuracy, it simplifies the calculation of the time to impact for approaching objects, and it can take up to 200 images per second, that it is sufficient for movement analysis. Also, we made use of a specialized pipelined processing architecture based on programmable devices that is able to calculate the time to impact at a rate of 80 images per second that is enough to detect car approaching movements of 100 Km/h or above.

Finally, this paper has shown how a retinal sensor can be used to effectively evaluate the time to impact on scenes where the different involved objects had considerable dimensions and speeds. This pretends to constitute a first approach for movement detection on navigation systems for collision prevention, even though there are still other areas that could take advantage of such an application.

References

[1] G.D. Hager, "Using resource-bounded sensing in telerobotics," in *Fifth Int. Conference on Advanced Robotics*, Pisa, Italy, June 1991.

[2] F. Pardo, B. Dierickx, and D. Scheffer, "Space-variant non-orthogonal structure CMOS image sensor design," *IEEE Journal of Solid State Circuits*, vol. 33, no. 6, pp. 842–849, June 1998.

[3] F.J. Blasco, F. Pardo, and J.A. Boluda, "A FPGA based PCI bus interface for a real-time log-polar image processing system," in *XIV Design of Circuits and Integrated Systems, DCIS99*, Palma de mallorca, Spain, Nov. 1999.

[4] J.A. Boluda, J.J. Domingo, F. Pardo, and J. Pelechano, "Motion detection independent of the camera movement with a log-polar sensor," in *13th International Conference on Digital Signal Processing, DSP'97*, Santorini, Greece, July 1997, pp. 809–812.

[5] J.A. Boluda, F.J. Blasco, F. Pardo, and J. Pelechano, "A pipelined reconfigurable architecture for visual-based navigation," in *Euromicro 99*, Milan, Italy, Sept. 1999.

[6] M. Tistarelli and G. Sandini, "On the advantages of polar and log-polar mapping for direct estimation of time-to-impact from optical flow," *IEEE Trans. on PAMI*, vol. PAMI-15, No. 4, pp. 401–410, 1991.

[7] K. Daniilidis, "Optical flow computation in the log-polar plane," in *Int. Conf. on Computer analysis of images and patterns, CAIP'95*, 1995, pp. 65–72.

[8] J.L. Barron, D.J. Fleet, and S.S. Beauchemin, "Performance of optical flow techniques," Tech. Rep. RPL-TR-9107, Dpto. of Computing and Information Science, Queen's University, Kingston, Ontario, July 1993.