Estimating time-to-impact data with an autonomous vehicle portable pipelined architecture

José A. Boluda, Fernando Pardo, Esther de Ves

Departament d'Informàtica. Universitat de València

Avda. Vicent Andrés Estellés S/N. 46100 Burjassot (SPAIN)

Phone: +34 96 3160 412/410 Fax: +34 96 3160 418 email: Jose.A.Boluda@uv.es

Abstract

The approach presented in this paper focus the problem of obtaining quickly time to impact data and extracting scene information from these data. The use of log-polar vision reduces the amount of data to be processed and simplifies the time to impact computation to a simple division of gradients and temporal derivatives. A custom hardware approach has been made for implementing a parallel processing pipeline with the aim of speeding-up the time to impact computation. Taking into account that the time-to-contact computation module is addressed to autonomous vehicles, the hardware implementation must be small and low-power demanding. The custom hardware module output will give time to impact maps that must be interpreted in order to obtain reliable scene information. Different objects placed at separated distances give different areas in the time to impact maps. These image sectors are a time to impact value distributions that present spatial neighborhood. Different times to impact areas are associated to different objects and can be classified through segmentation strategies.

1. Introduction

Sensor data fusion based models are strategies for allowing safe autonomous vehicle navigation in non-structured environments [1]. Time-to-impact computation form optical flow is an interesting algorithm useful for robotic navigation [2]. Avoiding collisions to static obstacles is a time-consuming task in non-structured environments. The accuracy needed for achieving a reliable navigation requires complex computations for obtaining precise data, and this requirement seems conflicting with the fast time reaction needed for an autonomous mobile vehicle.

In this way a custom or hardware approach can be interesting for solving real-time constrains. Programmable logic offers a wide spectrum of devices useful for implementing custom hardware from a high-level hardware description language.

On the other hand, space-variant vision emerges as a useful image representation, taking into account that information reduction is interesting for a system with hardware restrictions. Particularly, log-polar mapping shows, as a particular case of space-variant vision, interesting properties in addition to the selective reduction of information [3], [4]. Especially, the time to impact computation can be easily obtained from the optical flow map when the camera movement is along the optical axis [5]. Figure 1 shows the sensor plane and its relationship with the computation plane.

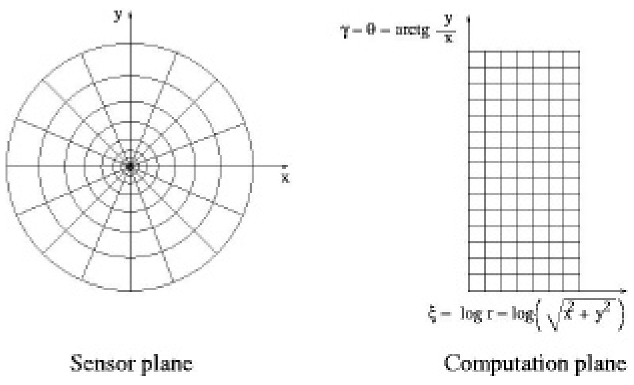
y
x
$\gamma = \theta = \mathrm{arctg}\,\frac{y}{x}$
$\xi = \log r = \log\left(\sqrt{x^2 + y^2}\right)$
Sensor plane
Computation plane

Fig. 1. Log-polar transformation

Time to impact computation from optical flow in the Cartesian plane is a complex task. The computation of optical flow is as reliable as time-consuming [6] [7]. Therefore, obtaining time to impact computation maps from the Cartesian optical flow can be an impossible real-time task.

Otherwise, if the sensor has a polar distribution, the movement can be assumed that only has a radial component. Moreover, if the sensor size and density follows an exponential law as the log-polar system coordinate does, it can be proved [5] that the time to impact value is proportional to the division between the radial gradient and first order temporal derivative. The proportionality constant is related with the sensor geometry.

It should be noted that this computation is simpler that its equivalent Cartesian. In this case only the gradient along the radial coordinate and the differences between two images must be taken into account. The gradient can be computed as the gray level difference between two neighbor pixels placed at consecutive rings. The temporal derivative analogously can be calculated as a simple difference between the gray levels of the same pixel into two successive images. This simple approach can be improved through the use of a window of pixels instead of two neighbor pixels for computing these differential magnitudes

[8]. In spite of this method for obtaining more accurate time to impact values, a statistical distribution appears. The statistical treatment of these distributions will determine the reliable presence of objects at different depths.

3. The pipelined computing approach

The log-polar representation used has been the transformation made by the CMOS log-polar sensor [9].

If the log-polar image acquisition system delivers a high ratio of images per second it will be possible to obtain a large quantity of time to impact maps. If the vehicle processor computes the radial gradient and the first order temporal derivative sequentially, the processing stage can be an unavoidable bottleneck.

Differential algorithms, as is the time to impact computation proposed, can be described globally as algorithms that apply several simple local differences to the entire image. Differential approaches use the temporal and spatial derivatives of the image sequence, as optical flow methods do, but they do not try to build explicitly the optical flow field. The meaning of local here includes the temporal dimension, so it is not reduced to spatial local differences, as is the gradient. The operations involved are simple and suitable for parallel implementation. These algorithms are also computationally intensive due to the image size, but they benefit from data reduction. These simple computations can be applied in parallel as a processing pipeline to the log-polar pixel stream supplied by the camera. In this way, a simple custom module that has log-polar images as input and time to impact maps as output has been implemented.

The processing module is a pipeline of three processing elements (PEs) that have an SRAM-based FPGA, two double port memories and a local memory [10]. The double ports memories are used for speeding-up the computation of temporal differences since are utilized as intermediate PE frame-grabber. Moreover, the pipelined take advantage of the two ports of these memories for accelerating the data flow.

The local memory is useful for storing intermediate computations since the FPGAs have only 820 flip-flops and most of them are needed for defining the PE functionality.

Each FPGA included into a PE has been designed with synthetisable VHDL, implementing a Finite State Machine (FSM) for communicating with previous and next PE, and for implementing its own processing functionality. The communication protocol among PEs follows a simple asynchronous protocol identical to all PEs. Moreover, this data-flow scheme is the same for obtaining log-polar images for the first PE of the pipeline and for supplying results to each stage.

The log-polar camera is connected to a frame-grabber that is controlled by the robot navigation system. The frame-grabber supplies log-polar images to the first PE of the pipeline. The PEs are interconnected with a plain parallel cable, and they are also connected to a common small rack, containing the common control and power supply lines. The last PE is also connected to the frame-grabber to receive results from the processing pipeline. Then the processed data are accessed by the vehicle processor, which implements the software stage as another running process.

The connection files that must be downloaded at each FPGA are generated once the synthetisable VHDL has been simulated for all the PEs of the pipeline. The autonomous vehicle stores these binary files and programs each PE individually through a PCI pipeline control/acquisition card.

A generic log-polar vision algorithm must be split into different well-balanced stages. The input to the first PE is a sequence of log-polar images, but the output of this first PE, and subsequently the inputs of any other PE, may be an ordered data structure.

The reconfigurable pipeline is being used for implementing several differentia image processing algorithms, between them the time to impact algorithm, which has been split into four processing stages. The nature of the three top stages suggests its implementation using the PEs described before since they are simple systematic operations easily formulated through hardware.

Figure 2 shows an overview of the time to impact pipelining implementation within the custom processing pipeline and the software interpretation stage.

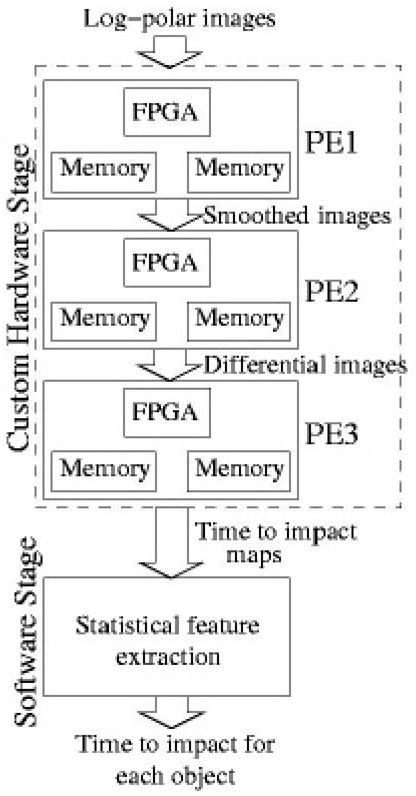

Fig. 2. Time to impact processing pipeline

- **Image smoothing**. It is not possible compute differential magnitudes both: if there are not differences between images or if there are gray level edges in the image. This smoothing has been made taking into account a three dimensions convolution mask for obtaining a medium gray level for each pixel.
- **First order differentiation and radial gradient computation**. This PE has as input smoothed images and computes gray level differences between:

- pixels at successive rings (radial gradient)
- and pixels at successive images (first order temporal derivative)

For making these computations they have been used the double port and local memories for storing different sequence images.

- **Time to impact map computation**. This third and last hardware stage divides both values for obtaining time to impact values, giving as output a time to impact value at each pixel. This stage makes an integer division of both values. More complex algorithms that allow floating point division did not fit within the FPGA of a single PE.
- **Time to impact map interpretation**. Ideally each object (if it is place perpendicularly to the camera position) should give the same time to impact value at its pixels. Unfortunately, due to the non exactly accomplishment of several algorithm constrains there is a time to impact statistical distribution around the real value. This non-achievement of the ideal circumstances is directly related with non-differentiable conditions at the log-polar images. Analogously, if the camera movement occurs non-along its optical axis the method will give meaningless values. In this way, an interpretation map stage must read the time to impact maps and extract reliable information about objects presence.

4. Obtaining time to impact data from log-polar sequences

Several experiments has been performed in order to identify which kind of statistical distributions will appear instead of a single time to impact value for a single object. Figure 3 shows a sequence where the log-polar camera has been mounted in a mobile platform, having a constant perpendicularly approaching movement to two objects.

These two objects are separated by 75 cm and initially the camera starts its movement with a distance of 75 cm from the first one. It should be note that the central white area corresponds to the central sensor region called fovea that does not follow the log-polar growth law expressed at figure 1 due to technological reasons. Since this zone does not conform this transformation the time to impact simplification assumed by the log-polar transformation does not apply, so these pixels have been eliminated.

The speed of the camera has been fixed to 5 cm/s and the image acquisition system will deliver 10 images per second. It should be noted that the system is able to deliver a higher image rate, but it makes no sense since two very close images will give negligible differences, making meaningless all the differential computations. This condition should be carefully taken into account since the special geometry of the log-polar sensor (big pixel size at the periphery) and the high image acquisition and processing rate could remove differences between two consecutive images.

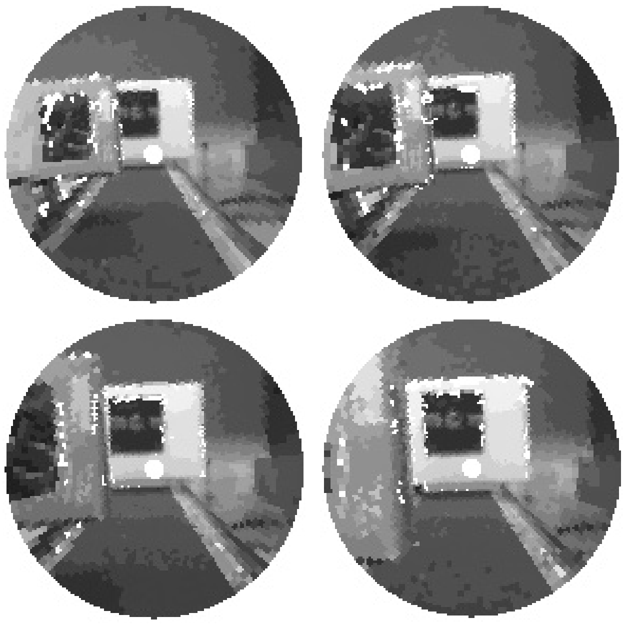
Fig. 3. Experiment sequence

There is a minimum acquisition interval between two images which guarantees well defined differences. This time interval has been analytically evaluated for implementing a motion detection differential algorithm with log-polar images at the reconfigurable pipeline [11], but it is useful for any differential log-polar algorithm. The acquisition interval depends of the camera velocity, sensor geometry and object gray levels.

The time to impact computation maps resulting from the sequence of figure 3 should give theoretically two constant values at each object surface, but as it has been appointed before, a statistical distribution appear for each object. Moreover, not all the time to impact values computed at each surface must be taken into account. It must be remarked that if the surface has a constant gray level no differences will appear both: at the radial gradient and at the temporal derivative. In this way, the time to impact must be computed when the differences exist and are well defined. Pixels that contribute to the time to impact value are marked at figure 3 as white pixels. These pixels belong to the object edges than once smoothed will give a constant gradient, useful for these differential computations.

Time to impact computation is very sensible to noise due to its differential nature. The gray level function I(ξ,θ,t) must be continuous and differentiable at all the points where the differential magnitudes are computed. Nevertheless the image is clearly a discrete function both, in time and in space. Despite of this source of noise the assumption usually adopted is to suppose that the gray level image is continuous if there have been any previous smoothing.

Moreover, inaccurate small temporal derivatives will give abnormally large time to impact values. Only differential magnitudes larger than an experimental threshold, thus with a small relative error, must be divided.

Figure 4 shows, as an example, time to impact histograms obtained from the last hardware stage for the images shown at figure 3.

The X-axis at each distribution represents the time to impact value and the Y-axis the occurrences for each time to impact value.

It can be shown the large dispersion that appear at the time to impact maps obtained from the hardware stage. Despite of these deficient initial results it is possible to see how there are two groups of values that will correspond to the two objects with the expected different time to impact.

The most far-away object presents less valid points for obtaining a reliable time to impact since the variations are smaller and thus the dispersion at the values will be higher. These points correspond to the small distribution placed at the right of the figures (higher time to impact value).

Alternatively, the closest object presents a great quantity of points and these points with less dispersion. These results are better since the closest object will present larger differences between images. Thus the relative error will be smallest and the computed value will more reliable.

The time to impact maps present a large dispersion and despite of the two Gaussian distributions that can be guessed from a visual first look, a deeper statistical study must be applied in the software stage.

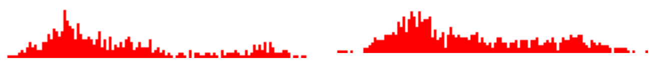

Fig. 4. Time to impact histograms corresponding to the experiment sequence

5. Conclusions

It has been presented a fast implementation of the log-polar time to impact computation algorithm. This implementation combines both: a custom hardware pipeline for accelerating the simple computation of the time to impact values at each pixel, and software stage that will extract the time to impact values from the statistical distributions that are the result of the hardware stage.

6. Acknowledges

This work has been supported by the *Ministerio de Ciencia y Tecnología* and FEDER under project TIC2001-3546.

References

[1] D. Fox, W. Burgard, S. Thrun, "The dynamic window approach to collision avoidance", *IEEE Robotics and Automation*, 4(1), 1997.

[2] N. Ancona, T. Poggio, "Optical Flow from 1-D Correlation: Application to a Simple Time-to-Crash Detector", *International Journal of Computer Vision*, Vol. 14, pp. 131-146, 1995.

[3] C. Capurro, F. Panerai, G. Sandini, "Dynamic Vergence Using Log-Polar Images", *International Journal of Computer Vision*, Vol 24(1), pp. 79-94, 1997.

[4] F. Jurie, "A new log-polar mapping for space variant imaging. Application to face detection and tracking", *Pattern recognition*, Vol. 32(5), pp. 865-975, 1999.

[5] M. Tistarelli, 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 15(4), pp. 401-410, 1991.

[6] J.L. Barron, D.J. Fleet, S.S. Beauchemin, "Performance of Optical Flow Techniques" *International Journal of Computer Vision*, Vol 12(1), pp. 43-77, 1994.

[7] S. S. Beauchemin, J. L. Barron, "The computation of optical flow", *ACM Computing Surveys*, Vol. 27(3), pp. 433-467, 1995.

[8] F. Pardo, J.A. Boluda, I. Coma, "High speed log-polar time to crash calculation of mobile vehicles", *Image Processing & Communications*, (To appear).

[9] F. Pardo, B. Dierickx, D. Scheffer. "Space-Variant Non-Orthogonal Structure CMOS Image Sensor Design", *IEEE Journal of Solid-State Circuits*, Vol. 33(6), pp. 842-849, 1998.

[10] J.A. Boluda F. Pardo, "A reconfigurable architecture for autonomous visual navigation", *Machine Vision and Applications*, (To appear).

[11] J.A. Boluda, J.J. Domingo. "On the advantages of combining differential algorithms, pipelined architectures and log-polar vision for detection of self-motion from a mobile robot". *Robotics and Autonomous Systems*. Vol. 37(4), pp. 283-296, 2001.