Concept of time-triggered image acquisition for embedded control applications

Wilfried Kubinger and Martin Humenberger
ARC Seibersdorf research GmbH
TechGate Vienna, A-1220 Vienna, Austria
{wilfried.kubinger,martin.humenberger}@arcs.ac.at

Abstract

In this work-in-progress report we present our current work on a time-triggered concept for image acquisition and processing for embedded control applications. We discuss some requirements for distributed embedded real-time control systems and its impact on visual sensor systems. We present a concept how we can achieve fixed latency and minimal jitter by handing the scope of control from the sensor to a VisionNode. Based on an example we illustrate the pro's – fixed latency and minimal jitter – and con's – higher latency compared to event-triggered control – of our concept. A short outline of our planned experimental setup for a platooning demonstrator is given. By these experiments we will evaluate our concept based on a real-world automotive application.

1. Introduction

This work-in-progress paper reports the current progress of our work on new concepts for time-triggered image processing. This work has been stimulated by the requirements of modern control systems, which are increasingly embedded in cars [2] or mobile robots [6].

For digital control algorithms used in distributed embedded computer systems, the following topics must be taken into consideration [1]:

- A time-invariant control algorithm expects a constant sampling period and a perfect periodicity. Otherwise the performance will degrade or the system will even get unstable, since the assumptions for the controller design are no longer valid [5].
- Real sensors have a delay between the acquisition of the physical entity and the reading of the measured and converted value by the computer system itself. If latency is inevitable it should be at least constant in order to minimize the jitter and to use a time invariant control law. The same problem exists with actuators.
- A small latency is preferred because sensor data might age rapidly and thus loose value simultaneously. Therefore the whole system will degrade, no matter if the delay is compensated by an advanced control algorithm or not.

This paper focuses on the acquisition of images using standard cameras by having an eye on the considerations given before. Vision is increasingly used as sensor input for closed-loop digital control, e.g. for controlling the movements of robots [5] or for keeping cars on the lane [3]. In these applications a stable control system with high performance is always requested.

In the following sections we present our ideas on introducing the time-triggered principle into vision systems and how we plan to demonstrate our concept. Furthermore we present first results and some already identified problems.

2. Why using time-triggered instead of event-triggered for image processing applications?

Classical image processing systems are event-triggered. That means, that an image sensor, usually a CCD- or CMOS-camera, periodically generates a sequence of images. The frequency of that sequence is e.g. 25 Hz for PAL cameras or 30 Hz for NTSC cameras. The disadvantage is that the sensor imposes the system a sampling rate. Furthermore, if the application program is not fast enough, images will be lost and a jitter is added to the sampling time.

The time-triggered paradigm [4], primarily developed for hard real-time applications, offers an excellent possibility to design and implement stable and powerful digital control systems [1]. These controllers can compensate the latency of a system (as long as it is fixed) and achieve a high dynamic by prediction. But it is necessary to have also sensors which can be integrated into a time-triggered environment. Up to now, work on image processing and pattern recognition focuses mainly on recognition algorithms but only little work has been focussing on the low-level vision system itself. In our work we deal with the image formation and transfer process and how this affects digital control loops [5]. Since more and more real-world applications become vision-based (e.g. driver assistance systems), it becomes necessary to investigate the interfacing issues between cameras and time-triggered systems.

3. Conceptional design

3.1. Overview

The principle of operation of an event-triggered image processing system, as it is used nowadays, is as follows. The camera sends the images in a sequential order in time and the frequency is usually defined by a video format like PAL or NTSC. Therefore the timing is not in the sphere of control of the system, it is defined by the sensor [4]. To achieve timely operation either the sensor must deliver a jitter-free and synchronized sequence of images or the system must trigger the sensor periodically to generate each new image. The first way is not practicable because no commercially available camera can generate such a stable and accurate signal in the time domain. Furthermore it is very likely to loose images from the camera due to the soft real-time design of that kind of system. This means that it is not guaranteed that the system has finished processing of image n when image $n + 1$ arrives. Clearly this is not what we want for an embedded digital control system. To achieve a low jitter operation of the camera it is necessary to control the timing of the camera by the distributed embedded control system, which has to have a precise clock or time information.

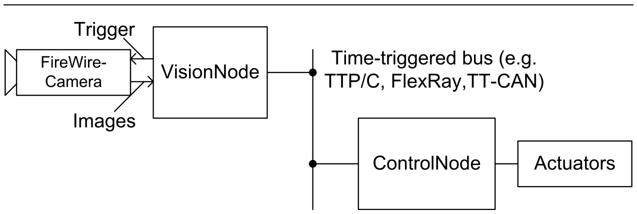

Figure 1. Layout of a vision-based distributed embedded control application

A time-triggered system is able to offer that kind of service. A system layout of this concept is given in figure 1. Due to the distributed nature of many embedded applications we split the work into two nodes, one responsible for the image processing task (the VisionNode) and one responsible for the control task (the ControlNode).

3.2. The operating sequence

A time-triggered system based on a TTP/C [4] or FlexRay [1] bus uses the TDMA (Time Division Multiple Access) principle, where each message is assigned to a periodic time slot. We exploit that behavior to precisely control the triggering of the camera and therefore the acquisition and the transfer of the images from the camera to the node. The signal timing between camera and VisionNode is shown in figure 2.

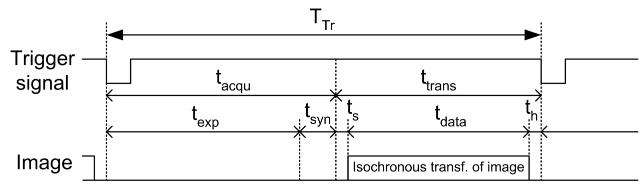

Figure 2. Signal timing between camera and VisionNode

In this figure the minimum required timing of an externally triggered image transfer is given. T_{Tr} denotes the trigger period, t_{acqu} gives the acquisition time, t_{trans} gives the transfer time, t_{exp} stands for the exposure time, t_{syn} stands for the synch time, t_s and t_h denote setup and hold time and t_{data} stands for the data transfer time itself. By the falling edge of the trigger signal the shutter of the camera is opened and the sensor cells of the camera collect the light. Depending on the brightness of the scene this time t_{exp} can vary widely. The synchronization time t_{syn} is necessary to synchronize the camera internal. Both times sum up to the acquisition time. In the following transmission period the image is transferred from the camera to the VisionNode. This behavior can be realized by using a FireWire-camera, which must support external triggering (e.g. SONY DFW-VL500 [8]). The transfer of the image data uses the isochronous mode of the FireWire-protocol, where a fixed bandwidth can be guaranteed [9]. As an outcome of figure 2, following relation

$$T_{Tr} \geq t_{exp}+t_{syn}+t_s+t_{data}+t_h \qquad T_{Tr} = const. \quad (1)$$

must hold for all circumstances to realize a jitter-free system.

It is important to note that, for a given camera and a fixed data-rate on the FireWire-bus, the maximum exposure time is limited by the set (or wanted) trigger period. This can lead to troubles when the application demands a long exposure time (e.g. dark outdoor scenes) but the control algorithm of the application uses a fixed sampling rate to achieve the needed performance. A possible workaround is to fix the exposure time and to use the iris of the camera for limiting the incoming amount of light. But this expects a sensor with a high light sensitivity and a special lens, both leading to a very expensive solution.

3.3. Parallel operations to optimize throughput

A good way to optimize the throughput of a system is to parallelize image acquisition and calculation of image processing algorithms [5]. Additionally new control signals are calculated in parallel with the transfer of the next image. Figure 3 depicts the temporal dependencies between camera, VisionNode, messages, and ControlNode. The acquisition of images is done by the camera. Then an image is transferred from the camera to the VisionNode, where the calculation of image processing algorithms takes place. The results of these calculations (e.g. new states) are used as input for the ControlNode, which generates new values for actuators.

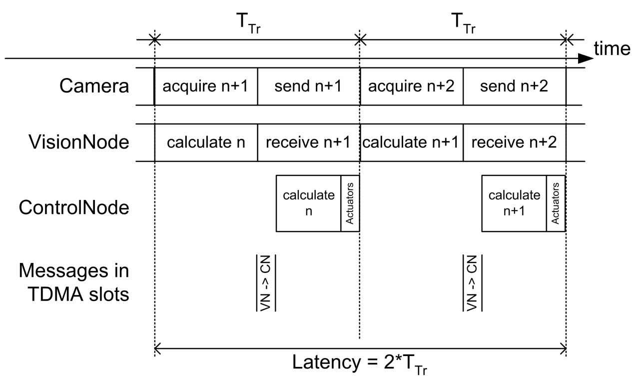

Figure 3. Parallelizing of image acquisition and image processing. n, n+1 – image numbers; VN – VisionNode; CN – ControlNode.

The sample rate T_{Tr} is given by the application and must also cope with the requirements of sensors and actuators of the system. The image acquisition frequency is $f_{Tr} = \frac{1}{T_{Tr}}$ and is under control of the VisionNode.

Using this time-triggered structure of image acquisition and processing, both latency and jitter of a system are determined. Since its time-triggered behavior, the jitter of the system is given by the quality of the clock-synchronization, TTP/C for example has a jitter in the range of microseconds [7], and by the ability of the camera to react to an external trigger signal. The latency of a vision-based control system is given by the time period between the start of the image acquisition and the end of processing and outputting the new control signals [5]. As shown in figure 3, the latency of the sensor system itself is less than two periods. Due to the distributed nature of time-triggered embedded systems it is possible to calculate the new control signals in parallel with the transfer of the new image. Therefore, and if the algorithm is fast enough, the latency of the calculation of the ControlNode can be covered within the latency of the vision system. The latency of the system is two unit delays and is fixed as long as equation 1 holds. In fact this means that the exposure time and the processing time of the algorithms at both nodes must be limited.

Due to the internal structure of the camera we do not have direct access to the internal timing of the camera. Therefore during startup of the system we have to establish a fixed relation in time between the camera and the VisionNode. We do this by sending out a single trigger signal, receiving the corresponding image and, after the hold time, start with the periodic trigger signal. This guarantees that the operation of the camera is in synch with the whole system. Furthermore we have to establish a re-synchronization after a specific period of time to avoid a divergency in time between the camera and the system.

3.4. Illustrative example

As an example we present the calculated latencies when using a SONY FireWire-camera DFW-VL500 [8] on a 400 Mbps FireWire bus. This camera is widely used in areas like mobile and service robotics. Table 1 presents a comparison of obtainable latency and jitter by using either the presented time-triggered or the classical event-triggered control of the camera.

	time-triggered	event-triggered
t_{exp}	25ms	parallel to t_{data}
t_{syn}	8.33ms	
t_s	1.9ms	1.9ms
t_{data}	30ms	30ms
t_h	1.4ms	1.4ms
T_{Tr}	66.66ms	33.33ms
Latency	133.32ms	99.99ms
Jitter	μs range	33.33ms

Table 1. Comparison of latency and jitter for time-triggered and event-triggered control of a SONY DFW-VL500 FireWire-camera with 640×480 pixels.

As depicted in table 1, one can clearly see that event-triggered control of a camera leads to less latency (in this case, the acquisition happens in parallel to the transfer, so the latency is three unit delays), but it is highly varying. This could be controlled by time varying control algorithms, but exact time-stamps of the image acquisition times are necessary. On the other hand, time-triggered control has got a higher but constant latency. This can be compensated by a time-invariant control algorithm.

4. Experimental setup

We use our concept of time-triggered image processing to serve as a sensor input for our demonstrator platform. This prototype is an experimental platform to show concepts of dependable embedded systems in real-world applications like driver assistant systems for automotive applications. A short outline of this platform for a platooning experiment, which is built up at the moment, is given in figure 4.

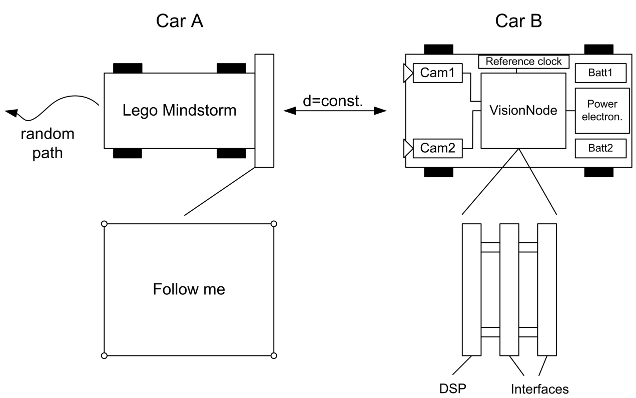

Figure 4. Experimental setup

The platooning experiment should work as follows. Car B follows car A by keeping the distance constant. This is usually called platooning [3]. Car B, our experimental platform, is equipped with a stereo camera system, an embedded real-time system (a VisionNode) and the power electronics for controlling the drives and the steering. The stereo camera system, which consists of two FireWire-cameras, is used to find and track the outlines of the ”follow me”-plate, which is mounted on the rear-end of car A. The vision system extracts the border of the plates, calculates the intersection points of the lines in each camera image and, finally, calculates the center point of the plate in both left and right camera image. By applying a triangulation algorithm the real position in camera coordinates (e.g. distance, azimuthal and polar angle) is calculated. These values are used to control the speed and the steering of car B to keep the distance constant and to follow the path of car A.

5. Conclusion

In this work-in-progress report we have presented our actual work on a concept for time-triggered image processing. We have discussed the problems, which arise by using state-of-the-art digital cameras and why constant latency and minimal jitter are requested by modern digital control applications. Furthermore we have presented how we can achieve fixed latency and minimal jitter by handing the scope of control from the camera to an embedded VisionNode.

We have given a short outline of our experimental platform, which is built up at the moment, and described the planned platooning experiments. By this demonstration we will show the combination of an embedded system and the time-triggered image processing hardware-concept to achieve a stable and reliable sensor input for control algorithms.

In addition we plan to analyze the interaction between the resulting performance and the latency of the vision system. To find the balance between the demand of high performance (means higher trigger frequencies) and a good outdoor capability (means higher exposure times) is an interesting research issue to be analyzed.

References

[1] Albert, A.: Comparision of Event-Triggered and Time-Triggered Concepts with Regard to Distributed Control Systems; *Proceedings Embedded World Conference 2004,* 2004.

[2] Fletcher, L., Apostoloff, N., Petersson, L., Zelinsky, A.: Vision in and out of Vehicles, *IEEE Intelligent Systems,* May-June 2003.

[3] Jones, W.D.: Keeping Cars from Crashing, *IEEE Spectrum,* Vol. 38, No. 9, September 2001.

[4] Kopetz, H.: *Real-time systems: design principles for distributed embedded applications;* Kluwer Academic Publisher, 1997.

[5] Krautgartner, P., Vincze, M.: Performance Evaluation of Vision-Based Control Tasks; *Proceedings of the IEEE International Conference on Robotics and Automation,* 1998.

[6] Murphy, R.R.: Rescue Robotics for Homeland Security; *Communications of the ACM,* Vol. 47, No. 3, March 2004.

[7] Poledna, S., Plankensteiner, M., Novak, M., Schlatterbeck, R.: Time-Triggered Architecture for real-time communication, *Embedded Control Europe,* February 2002.

[8] SONY DFW-VL500 Technical Manual, Ver. 1.0, *Sony Corporation,* 2001.

[9] Yoshimoto, H., Arita, D., Taniguchi, R.: Real-Time Image Processing on IEEE1394-based PC Cluster; *Proceedings of the 15th International Parallel and Distributed Processing Symposium,* 2001.