FPGA based approach for Heterogenous Sensors Data Fusion in Autonomous Vehicles

D.Créno, B.Senouci, R.Zitouni
Central Electronic Engineering School
ECE-Paris-Lyon
37, Quai de Grenelle, 75015 Paris, France
senouci@ece.fr

Abstract— *The new era of autonomous vehicles is considered as one of hot topics in Cyber Physical Systems exploration. It uses many sensors and functions to improve vehicle perception. The decision maker offers a flexible way to define the vehicle behaviour whereas the convoy driving mode is one important use case to explore more driving related issues. This paper reports an overview of on-going work on FPGA prototyping to improve the overall speed of the decision-making process and enable the convoy to drive at the higher speed safely. We present a data fusion methodology using heterogeneous Sensors (Lidar & Camera). Our methodology is based on an FPGA approach to speed up the processing time. A prototype has been built to explore new issues and solutions.*

Keywords— FPGA based design, RTOS, Smart Cars, AI, ADAS, Hardware accelerators, HW/SW Co-design

I. Introduction

During the last decade, Artificial Intelligence (AI) is revolutionizing the future of the car. The long-term goal is to offer high levels of autonomy. Driver assistance and driver awareness monitoring are among the fundamental steps towards full autonomy that will drastically increase safety on our roads. In addition, with the advent of better natural user interfaces the overall driving experience will be redefined. The autonomous car is defined as a vehicle capable of making all driving decisions on its own. They are therefore intelligent systems able to observe their environment and deduce the best reaction from the surrounding information. These systems are subject to numerous constraints, including safety or ecological requirements. As the users of these systems expect speeds at least equal to those currently reached by cars, the reaction time of the systems on board autonomous cars must be extremely short. This time constraint is commonly referred to real-time constraint. Because of this real-time constraint, it is necessary to optimize the speed of each component of the decision-making process in an autonomous vehicle as much as possible. The solutions to achieve this are varied, ranging from the optimization of decision-making algorithms to the modification of sensors. Indeed, FPGAs, as parallel architectures, have the promise of specialized hardware performance with high computational speed, lower clock frequencies and power consumption. The main objective of this paper is to build and prototype an ADAS (Advanced Driver Assistant System) based on the detection of a pedestrian in order to make an appropriate decision.

The rest of the paper is organized as follow: the second section is dedicated to the state of the art for pedestrian detection and for embedded systems design techniques. In section 3 we report our the HW/SW co-design of the system and our methodology. The section 4 shows the implementation and experimentation results, while the section 5 concludes the paper and presents our future works.

II. Approaches and Related Works

Silva et al. [7] have worked on the data fusion of LiDAR and camera sensor. They are with a similar computation to the one we will present for alignment of the data, but use this data differently afterwards. For example, they have evaluated two different free space detection algorithms on their data. Their prototype uses a similar vehicle, a wide-angle camera and a Velodyne LiDAR [3]. Number of research works have focused on fusion of LiDAR and camera data. A recent paper has shown how to perform data fusion and a depth estimation with object detection afterwards.[2]

III. Methodology

In this prototype we use different components like FPGAs, a Zedboard by Xilinx [1], a basic webcam by Logitech and finally a LiDAR Hokuyo URG-04LX-UG01 Scanning Laser Rangefinder.

FPGAs are a technology that consists of affixing a large number of multifunction logic circuits on a microchip. These circuits can be parameterized in their functions and interactions to create complex circuits. The fundamental differences with a classical processor are: A classical processor has a large number of logic circuits prepared and finalized from its conception; a processor uses successively these different circuits to perform tasks. Computer development work on a processor consists, at a fundamental level, of describing a sequence of uses of these prepared functions. Conversely, when developing on a FPGA, one works on describing an arrangement of logic components that will be reproduced on the electronic chip. The other major difference is that, once the logic circuit arrangement has been defined on an FPGA chip, it can no longer be modified without interrupting the process in progress, whereas on a processor new lists of functions to be executed (known as instructions) can be added. In exchange for this constraint, FPGAs obtain greatly increased efficiency compared to conventional processors (known as CPUs). The prototype uses the Linaro [4] operating system, which is based on Linux. In addition to this operating system, we developed the vehicle control system (VCS), which is the software that is running on the vehicle. The main unit is starting the threads for all the vehicle systems: WIFI communication, camera, LiDAR, decision maker. After all threads have been started and no errors occurred, the system is in manual mode and ready to start the driving process. This driving process is composed of different smaller process like steering process or servo motor process for example.

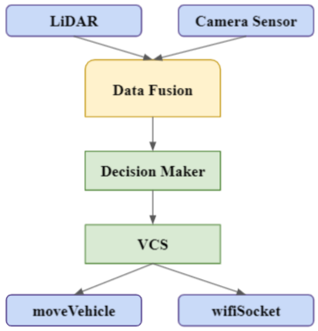

Fig 1: VCS Structure

Fig 1 shows from top to bottom: The LiDAR and Camera Sensor threads provide their data to the Data Fusion process which will merge the different data in order to obtain the more accurate data. Then, this data is provided to the Decision Maker. The Decision Maker is evaluating the data and transmits the result to the VCS main thread, where the result is stored and actions are sent to the moveVehicle threads (driving, steering) and to the WiFi Socket, which will transmit it to other vehicles. This prototype is thus composed of two cars running in convoy. When the "Leader" car detects an obstacle or a pedestrian, it stops or slows down and transmits the information to the "Follower" car so that the latter performs the same decisions.

A. Data Fusion

Data fusion is the process of integrating multiple data sources to produce more consistent, accurate, and useful information than that provided by any individual data source [6]. In the case of autonomous vehicles, the data of multiple sensors of different kinds can be combined. Once the two sensors are attached to the vehicle, it is possible to design a data fusion algorithm. The data is acquired, then a processing is carried out, after that a decision is taken by exploiting the result of this processing. The question of data fusion is therefore: Should we merge the data before processing and imagine a processing on the fusion of the data or rather merge the processed data? The pre-analysis data fusion solution has the advantage of being more portable for other sets of sensors, but imposes artificial intelligence as processing, and therefore requires training data set calibrated on the algorithm of fusion. This method therefore requires a lot of resources, in order to constitute a collection of training data for the artificial intelligence algorithm. Thus, we opted for the merge solution after data processing.

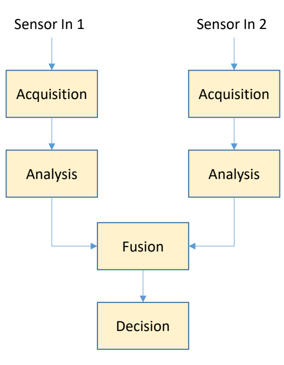

Fig 2: Data Fusion Scheme

After this decision we presented in Fig 2, the form that the data fusion algorithm takes in our particular case can be studied. Here, the goal of the algorithm is to determine a correspondence between the data provided by the LiDAR, a point cloud, and the image provided by the camera. The final data is a pixel-distance association. The problem posed is the following: How to determine, from the available information, the coordinates of a LiDAR point in the image? The problem can be broken down into multiple trigonometry problems. The final formulas make it possible to determine the theoretical X and Y coordinates in the image of the point perceived by the LiDAR. Once the mathematical formulas were determined, it was necessary to determine the precise algorithm that would be implemented on both FPGA and CPU.

```
Inputs: Camera field of view, lidar-Camera relative positions, image resolution, lidar input(=distance to lidar, angle from front).
Output: pixel – distance correspondence.
For each point in lidar input do:
    Compute corresponding pixel X coordinate
    Compute corresponding pixel Y coordinate
    If (X,Y) is in image resolution
        Compute distance to camera from distance
        to lidar and lidar-Camera relative positions
        add to list : [(X,Y), distance to camera]
    End if
End for
Return list
```

Fig 3: Data Fusion Algorithm

As shown in Fig 3, the Data Fusion Algorithm processes each point recovered by the LiDAR and associates, if there is a match, a distance with a pixel. Depending on the distance, there may be zero, one or more points per pixel. This algorithm has been tested and validated by first developing a version entirely on CPU, in C++. The program exploits the OpenCV library, a library specialized in image processing, installed for this purpose. The program takes a picture and performs a LiDAR scan, uses the data fusion algorithm and then a people detector algorithm. Combining these two results, we can determine the distance at which the pedestrian(s) photographed are pedestrian(s) is/are.

B. Decision Maker

The Decision Maker (DM) (Fig 4) is the central unit that computes whether the vehicle needs to stop or can continue to drive. The main goal is to minimize any damage that could happen to the vehicle or its surrounding, especially to people. To do this, the sensor data can be combined in various ways to provide different insights about the environment. For example, the people detection will give us information, if there is a person in the view. The person must have entered the area that is visible to the camera, which is in front of the car. This means if the vehicle continues to drive forward, it might collide with the person and hurt him or her. So, from this information only, the vehicle should decide to stop until the person disappears and the way is free again. Now, if this information is combined with the LiDAR data, the vehicle can compute the distance of the human. With this additional information, it can decide whether it needs to stop immediately, because the person is very close; or if it is safe to continue to drive some distance - probably at a lower speed until the person is closer or until it has disappeared from the vision.

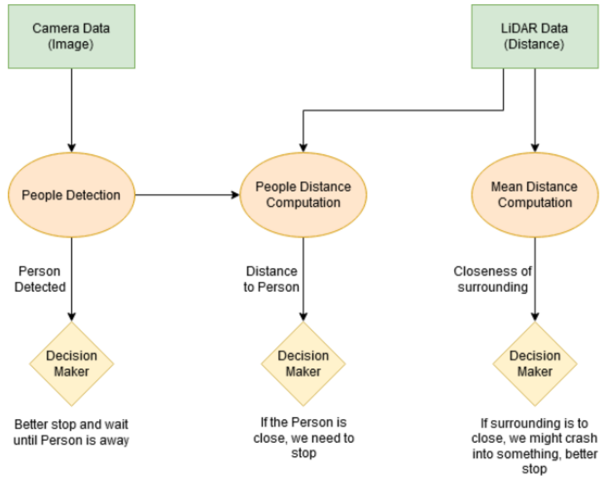

Fig 4: Decision Making Process

IV. Implementation and Experimentations Results

One of the main tasks was to make the detection process faster without losing detection accuracy. For that, different detection methods have been evaluated. OpenCV library offers a detection method that can be used: The People Detection and the Haar Classifier. By resizing the image to a smaller size (from 640×480 to 320×240), we reduce the contained information, but also, we reduce the amount of data that has to be processed.

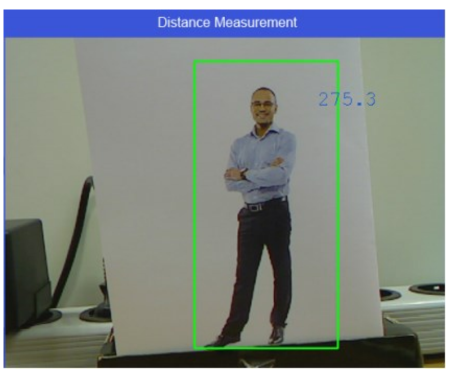

Fig 5: People Detection and Distance Measurement

The Haar classifier shows great improvements in computation speed. Furthermore, we remove small parts of the image on both sides, as it is not part of the vehicle driving path. For the Haar classifier, this improves speed further from 0,34s to 0,26s. In order to always improve the processing speed of our system we decided to do the processing of the Haar Classifier on FPGA. So, we used the Vivado HLS tool to transform the OpenCV C code into HDL code. With Vivado HLS we can transform the SystemC models to HDL. Currently, the pedestrian detection processing is done on the FPGA area. Testing the driving mode has shown that the vehicle control system is working well. Both vehicles follow each other perfectly and the decision maker makes stop when a pedestrian is detected in front of them (Fig 5).

V. Conclusion and Future Works

In this paper we presented a work in progress. We introduced our methodology of designing embedded systems and smart ADAS for Autonomous cars. The methodology integrates several steps. In the first phases of our work, we focused on the two main steps: Data Fusion and Decision Maker. The Data Fusion is done thanks to an algorithm fully implemented in software. This algorithm retrieves data from a Lidar and a camera and then merges the Lidar points with the camera images. This algorithm can therefore merge data from several different sensor sources. The Decision Maker allows to take a decision according to what it will have received as data. It can order the convoy to stop, slow down or continue according to the detected obstacles or pedestrians. All this is implemented on a prototype built from remote-controlled cars.

We continue investigating this topic by trying to propose a new HW/SW implementation of the main steps of the design. Future work may include the massive usage of the FPGA unit, which will speed up the decision-making process. At the current state, a function could be put on the FPGA: The data fusion, where the LiDAR point cloud data is combined with the image data. It is also possible to use more recent versions of OpenCV, which add more accurate and faster people detection methods, as well as the functionality to use pre-trained neural networks. In the future, the convoy followers may have sensors to simulate fully autonomous (from each other) vehicles. The goal is also to develop an adaptative data fusion algorithm able to deal with all types of data and then perform this data fusion on FPGAs. This data fusion must be able to retrieve any type of data, regardless of the sources of these data and regardless of their number. This multitude of data will then be merged to make the right decision. Finally, we are working on a new way to detect the different elements of the driving environment by using communication as sensors. Indeed, we try to use the Bluetooth to spot a traffic light and anticipate braking to reduce pollution. The data exchange between the cars will also be improved as we improve our WiFi transceivers.

References

[1] Digilent. ZedBoard Zynq-7000 ARM/FPGA SoC Development Board. URL https://store.digilentinc.com/ zedboard-zynq-7000-arm-fpga-soc-development-board/.

[2] G. Ajay Kumar, Jin Hee Lee, Jongrak Hwang, Jaehyeong Park, Sung Hoon Youn, and Soon Kwon. LiDAR and Camera Fusion Approach for Object Distance Estimation in Self-Driving Vehicles. Symmetry, 12(2):324, 2020. ISSN 2073-8994. doi: 10.3390/sym12020324. URL http://dx.doi.org/10.3390/sym12020324.

[3] Varuna De Silva, Jamie Roche, and Ahmet Kondoz. Fusion of LiDAR and camera sensor data for environment sensing in driverless vehicles. 2018. URL https://repository.lboro.ac.uk/articles/preprint/ Fusion_of_LiDAR_and_camera_sensor_data_for_environment_ sensing_in_driverless_vehicles/9466361

[4] Driving open source software development on Arm. URL https://www. linaro.org/abou

[5] Cyber-Physical Systems. URL https://ptolemy.berkeley.edu/ projects/cps/.

[6] D. L. Hall and J. Llinas. An introduction to multisensor data fusion. Proceedings of the IEEE, 85(1):6–23, 1997.

[7] Michelle Andrade Valente da Silva. SLAM and data fusion for autonomous vehicles : from classical approaches to deep learning methods. Machine Learning [cs.LG]. Université Paris sciences et lettres, 2019. English. ffNNT : 2019PSLEM079ff. fftel-03167034f