ILR - 09

Perception System Using Stereo Vision and Radar

Team A - Amit Agarwal

Harry Golash, Yihao Qian, Menghan Zhang, Zihao (Theo) Zhang

Sponsored by: Delphi Automotive

March 23, 2017

Table of Contents

1. Individual Progress

Since the previous progress review, I was tasked with working on the Radar and visualizing the tracking points received by the Radar through Ethernet. For this I worked with Harry and obtained data about tracking points from the Radar.

1.1 Delphi ESR 2.5

Over the past couple of weeks, Harry and I have been working on getting sensible data from the Delphi ESR 2.5 Radar. After multiple challenges to obtain data from the Radar and communication with Delphi, we were instructed by the team at Delphi to use the Ethernet port to get data instead. Previously, we were using the CAN connector to get data from the Delphi ESR 2.5 radar. The data obtained from the Radar through Ethernet is the raw detection level data in contrast to the processed tracking level data obtained through CAN. The usual procedure, as described in the previous ILRs, is used to connect the Radar to the test car.

The Radar is connected to the Laptop using through Ethernet to get the data. Through the terminal, the IP address of the computer is changed to on the same class of IP addresses as the Radar. The Radar uses a Class C IP address whereas most routers in today's world use Class A IP addresses. This makes it harder to get on the same network. After some in-depth research about networks, I could understand how to get on the same network as the Radar. Once this was done, data were obtained using the *netcat* command with port *5555* and IP address *192.128.2.21*. The data were obtained in hex format as shown in the previous ILR. Although this data was more understandable as hex, this was still far from the level of understanding needed.

According to Delphi, the correct form of interpreting the data is to parse it as hex values. These hex values are stored similarly to a CSV file. The data values of the information given from the Radar are stored one after the other without any labels in the form of a TCP packet. These hex values need to be parsed according a table supplied to us which helps in decoding each packet of data. The data that is received from the Radar is supposed to be shown in hex format.

To obtain the data, as TCP packets, from the Radar in a reliable and consistent manner in real-time Terminal commands were not enough and more research was required in the field of Socket Programming. After considerable reading about how TCP/IP and UDP work, how a packet is transferred as well as the significance of each step involved. Using the knowledge amassed through the literature review, I wrote code in C++ using boost libraries to obtain the data without using terminal. Once, the data was obtained continuously, the data needed to be converted to a human understandable format and not hex. This was done with the help of a document provided with the Radar, a part of which is shown in figure 1 on the next page.

g_look_index	Uint16	2	1	0	N/A	Scan index (normally updated every 25ms)
can_mmr_scan_index	Uint16	2	1	0	N/A	Scan index (updated every 50ms)
target_report_host_speed	Signed 16	2	1/128	0	m/s	Host speed
target_report_host_yaw	Signed 16	2	1/128	0	deg/s	Host yaw rate
g_XCP_timestamp	Uint32	4	1	0	N/A	Time stamp
Release_Revision	Uint8	1	1	0	N/A	DSP software version #1
Promote_Revision	Uint8	1	1	0	N/A	DSP software version #2
Field Revision	Uint8	1	1	0	N/A	DSP software version #3

Figure 1: TCP Parsing for the Radar

Further columns in the table above give the byte boundaries for the variables defined in the first column. These byte boundaries are used to extract the data and save them as variables in the program written in C++. This makes is easy to utilize these variables for manipulation, visualization and combining the Radar with the Computer Vision system. Once these variables were created and defined, a test was conducted to see the data in real-time. This was conducted by giving the parsed data to display in the Terminal in real-time. A screenshot of the output right before the *ctrl* + *c* is pressed is shown in figure 2 below.

```
DSP Version: 3.76.15
Scan type: 2
Scan index: 55746
Target count: 518
Target 1 range (m): 574

DSP Version: 3.76.15
Scan type: 1
Scan index: 55746
Target count: 516
Target 1 range (m): 1049

DSP Version: 3.76.15
Scan type: 2
Scan index: 55747
Target count: 6
Target 1 range (m): 565

^CCaught signal 2. Shutting down ESR 2.5 TCP Parser...
```

Figure 2: TCP Parsing for the Radar

2. Challenges

The major challenges faced were due to no previous knowledge of networking or socket programming for data acquisition from the Radar. In addition, the test car for the project has broken down.

2.1 Socket Programming

The knowledge possessed about Networks and Socket Programming by the team had been minimal until the Spring Break. This inhibited the team from approaching the Radar with some experience about the field. This was overcome by studying Socket Programming over spring break through online tutorials. More in-depth knowledge maybe needed at a future stage but that is highly unlikely.

2.2 Test Car

The test car used by the team, a Volvo S60 2.5T AWD, is not in working condition anymore because of hitting an animal on the interstate over spring break. The car should be repaired within a few weeks. This makes it a challenge to obtain real-world data with the car. To minimize this, some data may need to be collected using other cars.

3. Teamwork

The work this week was performed well by all 5 team members. Harry and I worked on the Radar and getting data from the Radar in real-time in a understandable and usable format. Yihao worked on implementing the object tracking code to work for multiple objects. Menghan and Zihao worked on the ROS module which will be used to integrate the Stereo Vision and Radar systems. All team members helped in setting up different parts of the recently arrived desktop computer. Everyone worked well and in a timely manner.

4. Future Plans

In the future, we plan to successfully visualize data from the Radar. On the Computer Vision side, the team will make progress towards an integrated CV system and eventually a complete integrated system including the Radar. I will be working on methods of combining the Radar data with the Vision data.

5. References

[1] "Delphi ESR 2.5". Delphi Support Center. N.p., 2017. Web. 12 Feb. 2017.
[2] "TCP/IP Network Programming". Vichargrave.github.io. N.p., 2017. Web. 24 Mar. 2017.