Faculty of Electrical Engineering & Computer Science
Master Degree Course Electronics Engineering, MSc EE

HSB
Hochschule Bremen
City University of Applied Sciences

Master 2017

Raviteja Gali

Control of a Safety System for Collision Detection of Autonomous Vehicle using Bumper Sensors.

ABSTRACT - Masterthesis

Safety is the major thing in driving a car. It is not in our control to stop the engine of a vehicle when an accident occurs. Due to extensive use of Robot Operating System (ROS) in advanced cars (especially in autonomous), this project is to simulate: when the car met with an accident; it should stop automatically in ROS environment.

The project was designed to drive electric vehicle system autonomously in an agriculture field. Objective of this thesis work is to: simulate the safety system for collision detection of autonomous vehicle using Bumper sensors. Car is surrounded by bumper. Electronic Control Unit consisting of a two push buttons are placed in: front and rare positions of car bumper. If the button is pressed (in real time scenario, occurring of crash) sends a CAN message which is received by the Receive Node on a ROS running system and transmits a CAN message to the microcontroller board: depending upon CAN message the LED on the board starts blinking (indicating in real time that engine gets stopped).

A ROS system is comprised of number of independent nodes; each node communicates with each other node (transmit & receive node) using publishing/subscribing messaging model. The receive node receive standard CAN messages from Microcontroller Unit (MCU) and sends a CAN message by a transmit node which makes LED on MCU blinks.

Writing a program- upon a push buttons signal sends CAN message to ROS node and from ROS it transmits a CAN message which makes LED on the MCU to blink through Peak CAN USB Interface. Creating peak CAN USB interface with ROS. Writing single purpose executable program: e.g. transmit node, receive node written in ROS client library; here the main client interface is roscpp which is C++ client library. Publishing standard CAN message format data over subscribed Peak CAN USB transmitted node.