eyeROBOT - Heterogeneous Robotic Platform for Real-time Computer Vision
Mohit Modi (mm2675), Sravya Chinthalapati (sc2655), Shang Dang (sd687), Yunong Liu (yl2494)
Advisor: Prof. Zhiru Zhang

ABSTRACT
Developing FPGA prototypes using hardware description languages (HDLs) is time consuming and limits the outreach of the designers. Instead, if the high level programming languages like C and C++ is used to develop hardware, then it can be developed much faster and can be adopted by wide range of programmers. This paper will design a heterogeneous design which can drive robot based on the analysis of real-time HD video stream on FPGA. The entire project is done using one of the most popular high level programming language C++, which opens a door of opportunity for software engineers to develop hardware prototypes without learning HDL languages. The purpose of the project is to learn the challenges and possibilities that high level digital synthesis brings during the development of the hardware.

iRobot Create is a mobile robot platform which can be driven based on the input commands provided over serial interface. In this project, irobot is programmed to drive in designated direction based on the movement of the object perceived from an HD video connected with FPGA. The project involves many aspects of the hardware-software co-design including FPGA prototyping. High-Level Digital Synthesis (HLS), Embedded System Development, Software Programming, Networking and Computer Vision.

IMPLEMENTATION
Implementation steps:
- Real Object is placed in front of HD camera to drive robot.
- HD camera takes the video and sends it to Xilinx ZC702 FPGA board over HDMI connection. Xilinx ZC702 board has built-in ARM Cortex A9 (1920 MHz) processor and reprogrammable FPGA fabric.
- ARM receives the input video frame and launches FPGA video accelerator by passing the video frame.
- FPGA fabric detects the object from video frames and outputs video and corner location of the object.
- ARM receives this data from FPGA, displays the video on output monitor, interprets the orientation of object from corners and sends command to iRobot over Ethernet based on object orientation.
- TP-LINK 5760 wireless access point broadcasts the data sent from FPGA.
- Intel Galileo microcontroller board has a wireless receiver (Intel N135 chip) which receives this data.
- Galileo then sends command to iRobot over UART communication channel at 57600 baud rate.
- iRobot interprets the command and drives by inferring the velocity and turning radius from the command.

Development Tools:
- [Xilinx SDKs + Xilinx Vivado Design Suite] To develop heterogeneous architecture for ARM and FPGA. It generates bitstream for FPGA fabric and Petalinux OS for ARM processor.
- [Xilinx ISE] Used to design / validate / debug Vivolog version of the FPGA video accelerator for comparison purpose.
- [Visual Studio + OpenCV] Used to develop software version of the project for performance comparison purpose
- [Arduino IDE] Used to program Intel Galileo microcontroller

RESULTS
As per the design, we are able to drive the robot successfully based on the object’s orientation in different directions. FPGA is successfully able to process HD frame at 60 FPS and able to send commands to the iRobot. Comparing the HLS design with Vivolog and software version, HLS stands in between both the designs. It’s far better than the software implementation of the project on Intel processor. The software only supports 13 FPS for SD video, whereas FPGA supports 60 FPS for HD video. Comparing HLS accelerator with the simulation of pipelined Verilog code demonstrates that the Verilog performance is slightly faster than HLS.

Heterogeneous Core

Acknowledgements
We would like to thank our advisor Prof. Zhiru Zhang for his constant guidance and encouragement, to Steve Dai for his support throughout the project and to Julie Wang for her detailed documentation explaining corner detection algorithm on FPGA. We are also grateful to Xilinx Inc. for providing SDK tools and numerous guidelines that have helped us in our project implementation.