GARDS: Generalized Autonomous Robotic Delivery System

Jade Zsiros  Brian Blalock  Darien Craig  Sudharsan Vaidhun  Alexander Wang†  Zhishan Guo
{jadezsiros, bblalock}@knights.ucf.edu, darien@craigs.us
sudharsan.vaidhun@knights.ucf.edu, WangA22@trinityprep.org, zsguo@ucf.edu
Department of Electric and Computer Engineering, University of Central Florida, Florida
†Trinity Preparatory School, Florida

Abstract—In this demonstration, we present a generalized platform customized to suit the needs of a fast power-efficient and autonomous delivery system. As an application demonstration, we deployed a mapping and localization system based on a combination of sensor sources. An online navigation algorithm utilizes the map information to deliver to a destination in the mapped area.

Index Terms—autonomous robots, GPU, embedded system

I. INTRODUCTION

The past decade has seen significant progress in energy-efficient computational hardware and machine learning techniques. In this work, we demonstrate our platform that combines these technological advancements for applied research in autonomous delivery vehicles. One of the predominant sensory inputs for autonomous vehicles is in one way or the other, related to visual information. The visual information of the surroundings can be sampled using a combination of camera, LiDAR, radar sensors. The parallel nature of processing these sensor data can benefit from their implementation on a GPU. While GPU operations are typically power-hungry, the availability of embedded low-power compute modules such as the NVIDIA Jetson TX2 with a peak power consumption of 15 Watt is suitable for our needs. Details of the hardware setup is described in Section II-A.

Besides having a good trade-off between power consumption and computational capability, a key feature of the platform is the general nature of the implementation. The following are the benefits of the platform for delivery robots

• **Capability to implement machine learning inference algorithms.** The machine learning algorithms need to modification from a desktop implementation due to the availability of general-purpose GPUs onboard.
• **Offline computation for global path planning.** The offline computation eliminates the need to be in constant communication with a database server.
• **Low power consumption.** Efficient computing platform saves power and regenerative braking to improves battery longevity.

II. DEMONSTRATION

A. Hardware

GARDS is built on the F1/10 platform [6]. The mechanical chassis models an all-wheel-drive (AWD) system and unlike differential drive platforms which are suitable for smooth indoor terrain, GARDS includes a suspension system for navigation on minimally structured terrain. The computational core of the platform is the NVIDIA Jetson TX2 [5]. The Jetson module has a quad-core ARM CPU, a dual-core NVIDIA denver CPU and a 256-core Pascal GPU capable for CUDA computing. The sensor system includes a 270-degree LiDAR, a stereoscopic ZED camera [8] and inertial measurement units. A snapshot of the platform integrated with the sensor system is shown in Figure 1.

B. Software

The software architecture used in the platform is the Robot Operating System (ROS) [7] and it is installed on a linux based Ubuntu distribution with a linux kernel modified for tegra devices such as the Jetson TX2. The ROS framework follows a publisher-subscriber model for inter-process communication among the various processes in the system. The stereo-camera
Fig. 2. Occupancy grid with dark region shown obstacles and inaccessible areas. 10 pixels of the occupancy grid roughly equals a distance of 1 foot.

Fig. 3. Constructed map of the cost function showing the cost to destination. The destination is represented with a red star.

publishes two video streams and an internal IMU data for frame of reference.

C. Navigation

The sensory information from two sources are integrated to for mapping, localization and navigation. First, the video streams from the camera are interpreted by the ZED software package into a depth image. RTAB mapping package [4] combines the information from the stereo camera and the LiDAR to construct an occupancy grid. A sample occupancy grid is shown in the Figure 2.

With the occupancy grid, a graph search algorithm such as D* is applied to create cost map similar to the map shown in Figure 3. The navigation software is custom written for this platform, using a graph search algorithm to create a routing gradient over the occupation grid for later vector navigation. As internal building delivery only needs to navigate to the doors inside the building, the navigation system generates one cost map for each end point, and stores the map for later use, cutting down on compute time while driving. With the occupancy grid relatively stable and unchanging, the construction of the cost map takes only a few seconds onboard. As an example, for an area of 5000 sq.ft. the cost map construction takes less than 15 seconds. The robot then loads the map into memory at the beginning of navigation, and then computes the drive vector from the map gradient at the robot’s current location, as reported by RTAB’s localization system. The localization system runs continuously while driving to keep the algorithm updated with the current position and orientation of the vehicle. The local obstacle avoidance while navigation is handled by the LiDAR.

III. Future Works

Currently, we have only exploited the general-purpose computing units in the GPU for processing video streams. Since the platform also has a varying-speed multi-core CPU cluster, we plan to implement energy-efficient multi-core scheduling algorithms [1]–[3]. Such scheduling algorithms bound the latency while also optimizing resource utilization and energy consumption. We expect to quantitatively measure the benefit of variable speed processors for application in autonomous mobile robots.

ACKNOWLEDGMENT

This work is supported by NSF grant CNS-1850851.

REFERENCES