A Stixel-Based Stereo Perception for Multi-Robot Systems

Narsimlu Kemsaram
Department of Computer Science, Multi-Sensory Devices (MSD) Group, Faculty of Engineering Science, University College London (UCL), 169 Euston Road, London NW1 2AE, United Kingdom
n.kemsaram@ucl.ac.uk

Anweshan Das
Department of Electrical Engineering, Mobile Perception Systems (MPS) Lab, Signal Processing Systems (SPS) Group, Eindhoven University of Technology (TU/e), 5612 AZ Eindhoven, Netherlands
anweshan.das@tue.nl

Gijs Dubbelman
Department of Electrical Engineering, Mobile Perception Systems (MPS) Lab, Signal Processing Systems (SPS) Group, Eindhoven University of Technology (TU/e), 5612 AZ Eindhoven, Netherlands
g.dubbelman@tue.nl

Autonomous driving is a very active research area, with virtually all major automotive manufacturers competing to bring the first autonomous car to the market. This race led to billions of dollars invested in developing novel sensors, processing platforms, and algorithms. In this paper, we explore the synergies between the challenges in self-driving technology and the development of navigation aids for autonomous robots. We aim to leverage the recently emerged methods for self-driving cars and use them to develop autonomous robots. In particular, we focus on perceiving the environment in real-time using stereo cameras. Then, as of proof-of-concept, we build a stereo perception framework for multi-robot systems based on a hardware platform used in the automotive industry. To perceive the environment, we adapt the implementation of the stixels algorithm. We discuss the challenges and modifications required for such an application domain transfer. Finally, we evaluate the framework on a cooperative automated vehicle research platform to show its usability in practice. The framework runs on a PC (x86_64 architecture) with a GPU card and the cooperative automated vehicle research platform with Drive PX2 (aarch64 architecture) compute platform.

I. Introduction

Robots primarily rely on camera-based systems to perceive and explore the surroundings. Specifically, in Multi-Robot Systems (MRS), individual robots must perceive the environment accurately. Moreover, multiple heterogeneous robots with individual sensors, algorithms, and hardware must act efficiently in performing complex tasks, such as urban platooning [1], search-and-rescue [2], disaster response [3], defense emergence [4], and space exploration [5]. These robots depend on highly accurate and efficient multi-sensor systems to perform these tasks. Sensors like Cameras, RADAR, LiDAR, Ultrasonic, IMU, etc, are widely used. Accurate 3D representation of the environment plays an important role in understanding and navigating around it. Sensors like LiDAR, Stereo Cameras, etc., are used to represent the environment in 3D point cloud format. These point cloud representations generally consume a lot of data, and processing these in real time is challenging. Stixels [6], are an efficient way to represent the environment in 3D.

In this paper, we present a prototype of stixel-based stereo perception framework for multi-robot systems and test its effectiveness in the real world environment. Figure 1 shows the block diagram of the stixel-based stereo perception framework for cooperative automated vehicles (CAVs), which can be further extended multi-robot systems like Unmanned Aerial Vehicles (UAVs), Autonomous Ground Vehicles (AGVs), Autonomous Space Rovers (ASRs), etc. The framework uses the left and right image frames from the stereo camera to rectify and compute a disparity map. It uses Deep Neural Networks (DNNs) to perform obstacle detection on the rectified image frames of the stereo camera. Then it estimates the depth of the detected obstacles using the computed disparity map. It also computes the radial distance, relative velocity, azimuth, and elevation angles of the obstacles based on motion estimation with respect to the ego vehicle. These DNNs are computationally expensive and require very powerful hardware to run. Recent advances in Graphical Processing Unit (GPU) technology and embedded hardware have made it possible to run these

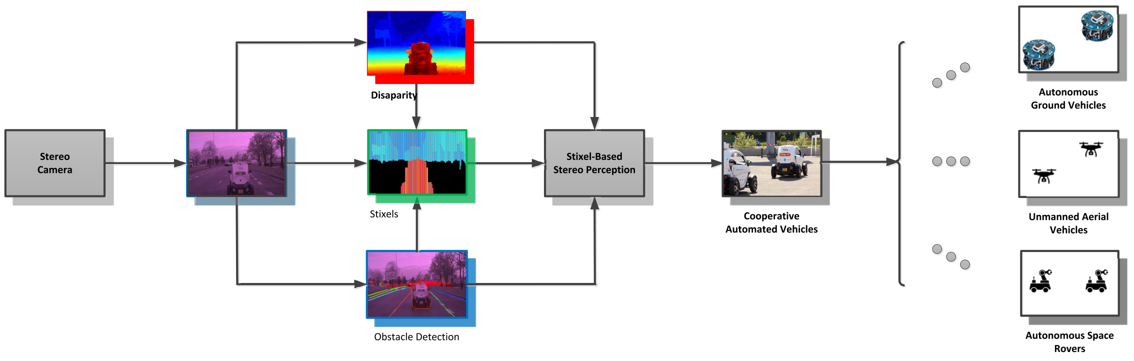

Fig. 1 Block diagram of the proposed stixel-based stereo perception for multi-robot system.

DNNs in real-time in a power-constrained environment like an autonomous vehicle. It sends the perceived obstacle information to the vehicle control system of the ego vehicle. It displays the perceived obstacle information through the Human-Machine Interface (HMI). The proposed perception framework is part of the integrated Cooperative Automated Vehicles (i-CAVE) research project. The main goal of the i-CAVE research project is to address the challenges related to automated and cooperative driving systems [7]. It consists of dual-mode vehicles, which can be driven autonomously and manually to allow maximum flexibility in urban mobility environment. To summarize, the key contributions of this paper are:

- The stixel-world implementation in [8] has been extended to work with a custom-built stereo camera on the automotive-grade hardware platform.
- The stixel-based stereo perception framework has been implemented and integrated into a single pipeline, which can detect and classify the objects, compute the stixels, identify the lane markings, and recognize the freespace boundary in front of the vehicle.
- The proposed framework has been evaluated and demonstrated with the cooperative automated driving research platform in a real-world scenario.
- We discussed and highlighted the potential areas that can be re-used in the future.

With these contributions and features, the stixel-based stereo perception framework will be applicable to wide use cases, including academia and industry research on the basics of the cooperative autonomous vehicle, multi-UAV system, and multi-rover system algorithms. The sections of this paper are organized as follows: Section II presents related work on the stereo vision, stixel-world, software frameworks, and hardware platforms. Section III discusses the concept of the proposed stixel-based stereo perception framework. In Section IV, we then present the experimental setup and results demonstrating the usability of our framework. The adaptation of our framework from cooperative automated vehicles to multi-robot systems is discussed in Section V. Finally, in Section VI, we present the paper's conclusions.

II. Related Work

The autonomous vehicle must perceive the environment using sensors, then the perceived information must be processed to understand the environment using perception algorithms, and finally, the processed information must be sent to the vehicle control system to generate a control command to the vehicle to move autonomously. Similarly, the robots must also perceive the environment using sensors and understand the environment using perception algorithms. However, instead of controlling the vehicle, robots must communicate with humans or other robots to alert them about obstacles in the environment. Although both tasks do not have identical goals, robots can benefit from the impressive research developments achieved in autonomous cars. Here, we present an overview of the current developments in stereo vision, the stixel-world, software frameworks, and hardware platforms of autonomous cars that can be applied to autonomous robots.

A. Stereo Vision

Stereo vision is widely used in various fields of applications such as advanced driver assistance systems, cooperative automated vehicles, multi-robot systems, multi-UAV systems, and multi-rover systems. *Stereo vision* is often used for object detection, lane detection, free space detection, and depth estimation. It can be used to generate a three-dimensional representation of the scene [9], which is utilized to estimate the position and motion of the detected objects [10]. The *stereo vision* uses a pair of cameras pointed at the same scene, and the differences between the images are used to extract the disparity data. Disparity estimation is a difficult task because of the high level of ambiguity that often appears in real situations [11]. The state-of-the-art algorithms of stereo disparity estimation are classified into three categories: local, global, and semi-global matching algorithms [12]. Local stereo vision matching algorithms are computationally less demanding which allows for real-time implementations [13]. The main disadvantage is that these algorithms are not able to handle featureless regions or repetitive patterns. The global stereo matching algorithms show a significant improvement in the disparity map quality, but it requires large computation power and is not applicable for real-time applications [14]. In 2005 Hirschmüllcr proposed the Semi-Global Matching (SGM) algorithm as a new alternative that achieves high-quality disparity map results while maintaining a reduced execution time [15]. Several real-time implementations were proposed based on the SGM algorithm using a Digital Signal Processing (DSP) unit [16], and Field-Programmable Gate Array (FPGA) [17] platform, with limited support for image resolution. With the advancement of GPU-based automotive platforms like Nvidia Drive PX2, Jetson TX2, Xavier NX, AGX Xavier, and Drive AGX, the execution of real-time stereo computation is possible [18] in power constrained environment such as an automated vehicle. In addition, depth cameras have been used in various applications, such as Intel RealSense D435 Camera, StereoLabs ZED Stereo Camera, and GMSL Camera. In this paper, we use the *SGM algorithm* with a custom-built *Gigabit Multimedia Serial Link (GMSL) stereo camera* on the Nvidia Drive PX2 platform.

B. Stixel-World

The *stixel-world* algorithm is an efficient method to generate a compact medium-level representation of the geometry of a traffic scene. The initial method analyses only the first row of obstacles in a scene [6], whereas a subsequent second version models the entire scene within the field of view [19]. The core principle is that it fits vertically stacked, piece-wise planar ground or obstacle segments to the input disparity data in a column-wise fashion. This process is cast as a maximum a posteriori (MAP) estimation problem that incorporates world knowledge to avoid physically unlikely situations (such as floating obstacles) and can be efficiently optimized using dynamic programming. The introduction of the *stixel-world* principle inspired new investigations, leading to extensions and improvements of the algorithm on different aspects. From a computational point of view, researchers have presented strategies to avoid full disparity estimation [20] or executing the algorithm on a GPU, leveraging the highly parallel nature of the stixel computation [21]. Similarly, the robustness of the *stixel-world* algorithm has been addressed by incorporating disparity confidences [22] or color data [23] into the analysis. Additionally, the representation has been enriched with more accurate geometry modeling [24] or semantic information [25]. As a result, researchers are increasingly incorporating stixel processing in larger systems, exploiting the compact representation for object detection [26], to train neural networks for freespace segmentation while driving [27], or for collision warning [28], and many others. Especially this last development, where stixel processing is integrated in framework for real-world deployment, is of interest to the work in this paper. Real-world deployment requires analysis on how the *stixel-world* algorithm can be leveraged in a wider system context, in parallel to other processes and in compute-constrained environments. To this end, this paper presents *a stixel-based stereo perception framework*.

C. Software Frameworks

Many automotive companies involved in self-driving platform development rely on open-source software frameworks. *Autoware Auto* is a Robot Operating System (ROS)-based open-source software, enabling self-driving mobility to be deployed in open city areas [29]. It provides a complete software stack for self-driving technology. *Baidu Apollo Open Platform* provides an open, reliable, and secure software framework to develop autonomous driving systems through on-vehicle and hardware platforms [30]. *Comma AI OpenPilot* is open-source software built to improve the existing driver assistance in most new cars on the road [31]. The *NXP BlueBox Automated Drive Kit* is a development platform series that provides the required performance, functional safety, and automotive reliability for engineers to develop self-driving cars [32]. The *Nvidia DriveWorks Software Development Kit (SDK)* is the foundation for all autonomous vehicle software development [33]. It provides an extensive set of fundamental autonomous vehicle capabilities, including processing modules, tools, and frameworks required for advanced autonomous vehicle development. It is modular, open, and designed to be compliant with automotive industry software standards. In this paper, we use the *Nvidia DriveWorks SDK software framework*.

D. Hardware Platforms

Deep learning algorithms are very demanding in terms of processing power. The powerful computing hardware is essential for autonomous vehicles to meet the computational demands of deep learning algorithms. Several suppliers are currently offering plenty of computing hardware platforms with different designs that show up on autonomous vehicles based on Application-Specific Integrated Circuit (ASIC), Digital Signal Processor (DSP), Field Programmable Gate Arrays (FPGA), and Graphic Processor Unit (GPU). *MobileEye EyeQ5* is the leading ASIC-based solution to support autonomous vehicles, and it provides 24 TOPS (Trillion Operations Per Second) computation capability with 10W power consumption [34]. *Google TPU v3* is the latest ASIC-based Google's AI accelerator for neural network and machine learning, and it provides 420 TFLOPS computation capability with 420W power consumption [35]. *Texas Instruments' TDA* provides a DSP-based solution for autonomous driving with less power consumption [36]. *Xilinx Automotive Zynq UltraScale+ MPSoC ZCU104* is an FPGA-based scalable solution that claims to deliver the right performance/watt for autonomous vehicles [37]. *Habana Labs' TPC* provides an FPGA-based solution to support deep learning workloads for autonomous vehicles with less power consumption [38]. *Nvidia Drive PX2* is a GPU-based powerful computer for autonomous vehicles, and it provides 24 TOPS computation capability with 250W power consumption [39]. *Nvidia Drive AGX Pegasus* is the newest solution, and it is ten times more powerful than the Drive PX2 platform. It is capable of 320 TOPS of processing performance with 500W power consumption [39]. In this paper, we use the *Drive PX2 embedded hardware platform*.

III. Proposed Stixel-Based Stereo Perception for Cooperative Automated Vehicles

For a long time, robot navigation has been limited to classical vision-based systems like CAVs [40], UAVs [41], and ASRs [42]. Autonomous vehicle navigation is clearly a more challenging task to deploy in a collision-free and safe environment. Therefore, extensive work has been developing perception aids to increase the vision range.

A. System Concept

The proposed framework is designed and developed for cooperative automated vehicles. In this work, we consider one lead vehicle and one ego vehicle. The lead vehicle is manually driven by the driver, and the ego vehicle autonomously follows the lead vehicle. The proposed framework is deployed on the ego vehicle, which continuously perceives the traffic environment from an on-board front-facing stereo camera. It processes the perceived traffic environment information and updates the vehicle control system of the ego vehicle in real-time. The vehicle control system computes a control command based on the processed traffic environment information. It provides computed control command to the actuator, such a way that the ego vehicle follows the lead vehicle autonomously. Also, it visualizes the perceived traffic information through the in-vehicle display in real-time.

B. Architecture Design

The proposed framework's architecture design is divided into two parts, namely software framework and hardware platform, which are explained below.

1. Software Framework

The proposed software framework consists of six parts: *image acquisition*, *stixel-world*, *deep neural networks*, *motion estimation*, *send output messages*, and *render outputs*. The first part is an *image acquisition*, which acquires the synchronized images from the stereo camera for the *stixel-world* processing algorithms. It also pre-processes the input raw images to a desirable format of processing algorithms. It reads a raw image frame from a stereo camera sensor, and it converts the frame into an NVMedia type image using Image Signal Processor (ISP). The NVMedia image is an Nvidia proprietary framework that uses dedicated hardware blocks on the Tegra processor for faster image processing. The stereo vision expects an image with linear memory. However, NVMedia has a block layout. For this reason, the image is converted to CUDA image format using 'Get CUDA (RGBA) from NvMedia (RGBA) Image Format'. It also reads the rig configuration file (JSON format) containing camera calibration parameters. The second part is a *stixel-world*, which computes the stixels using stereo vision and extended multi-layer stixel algorithms. The stereo vision uses the camera calibration parameters acquired from the rig configuration file to undistort and rectify the input stereo image pairs. The rectified stereo image pairs are used to compute the disparity map. For stixel computation, the disparity map image CUDA (RGBA) is converted to OpenCV (Mat) image format to suit the extended multi-layer stixel algorithm needs. After stixel computation, the output OpenCV (Mat) image is converted to a CUDA (RGBA) image format to suit the

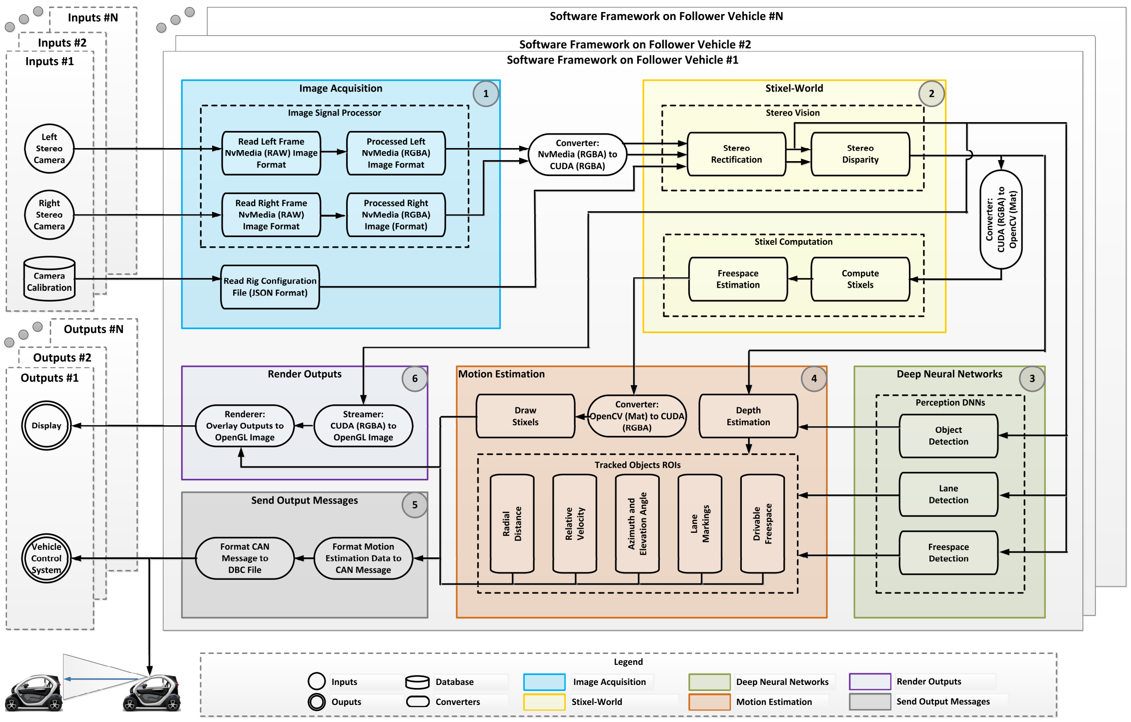

Fig. 2 A stixel-based stereo perception software framework for cooperative automated vehicles: (1) Image acquisition, (2) Stixel-world, (3) Deep neural networks, (4) Motion estimation, (5) Send output messages with motion estimation data, and (6) Render outputs.

renderer's needs. The third part is *deep neural networks*, which computes the perception algorithms such as object detection, lane detection, and freespace detection. The object detection classifies and tracks the detected objects and provides the Region Of Interests (ROIs) to the *motion estimation*. The lane detection identifies and classifies the lane markings and provides the identified lane markings to the *motion estimation*. The freespace detection recognizes and classifies the drivable freespace in front of the vehicle and provides the recognized freespace boundary to the *motion estimation*. The fourth part is a *motion estimation*, which takes the tracked objects's ROIs from the object detection and disparity map from the stereo disparity. It computes the depth of each tracked object. It estimates the object motion parameters such as radial distance, relative velocity, azimuth angle, and elevation angle based on the tracked objects' computed depth. The fifth part is to *send output message with motion estimation data*, which formats the motion estimated data using the database container (dbc) file and updates the vehicle control system over the CAN bus. The CAN message from the proposed framework, which corresponds to the object motion parameters, are decoded using the dbc file by the vehicle control system and generates a control command. The sixth part is *rendering outputs*, which overlays the estimated motion data on the processed image that is visualized on an in-vehicle display over an HDMI cable. The outline of our proposed perception's software framework is shown in Figure 2.

2. Hardware Platform

The hardware components used for the proposed framework are the DC-DC power converter, the Drive PX2, a custom-built Sekonix GMSL-based stereo camera, a vehicle control system and an HDMI display. The DC-DC converter is installed onto the 24V DC battery of the Renault Twizy, which converts to 12V DC, and it is used to power the Drive PX2, stereo camera, and display. The stereo camera uses the GMSL protocol and is plugged into the Drive PX2 GMSL camera ports. The GMSL cameras are chosen for their compatibility with the Drive PX2 development environment. The Drive PX2 hardware platform interfaces the stereo camera, executes the stixel-based stereo perception framework, and communicates with the vehicle control system. The proposed framework and the vehicle control system communicates via CAN bus. The proposed framework visualizes the perceived traffic information through the in-vehicle HDMI display in real-time. The outline of our proposed framework hardware platform is shown in Figure 3.

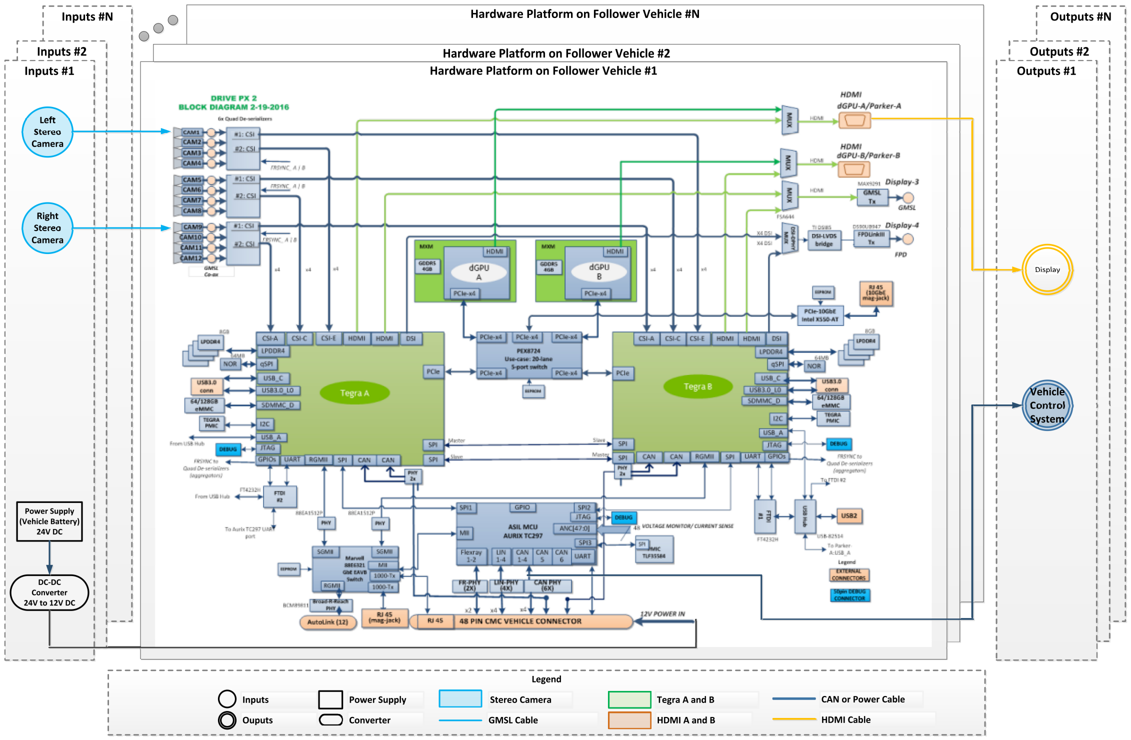

Fig. 3 A stixel-based stereo perception hardware platform for cooperative automated vehicles [18].

C. Development

The developed modules such as stereo vision, stixel computation, object detection and tracking, lane detection, and freespace detection, are explained below.

1. Stereo Vision

The acquired left and right synchronized input images from the stereo camera are used for stereo rectification and disparity computation. The stereo rectification algorithm runs on the GPU and generates rectified images based on the stereo camera calibration parameters. The stereo disparity algorithm takes the rectified images and runs on the GPU using CUDA programming. It computes a disparity map using a stereo Semi-Global Matching (SGM) algorithm [43].

2. Stixel Computation

The employed implementation of the stixel computation sub-module is derived from the original multi-layered stixel-world algorithm [8], and includes the extensions as presented for the disparity-baseline system discussed in [23]. It extracts the stixels from the input disparity map, and it allows for multiple stixels along every column.

3. Object Detection and Tracking

The object detection and tracking sub-module uses Nvidia's proprietary DNN called *DriveNet* [44] to perform real-time object detection and tracking. It consists of three parts, such as object detection, object clustering, and object tracking. The input to the object detection algorithm is the RGBA image format, and the output is object proposals with bounding boxes. Each object can have multiple proposals. The object clustering algorithm clusters these various proposals into one bounding box for each detected object. The object tracking algorithm tracks the detected bounding boxes to maintain temporal consistency. The output of this algorithm is the bounding boxes and class labels of the detected objects. It detects and classifies five different classes of objects: car, pedestrian, bicycle, traffic sign, and traffic light. The color of the bounding boxes represents the classes that it detects: red for cars, green for traffic signs, blue for bicycles, magenta for traffic lights, and orange for pedestrians.

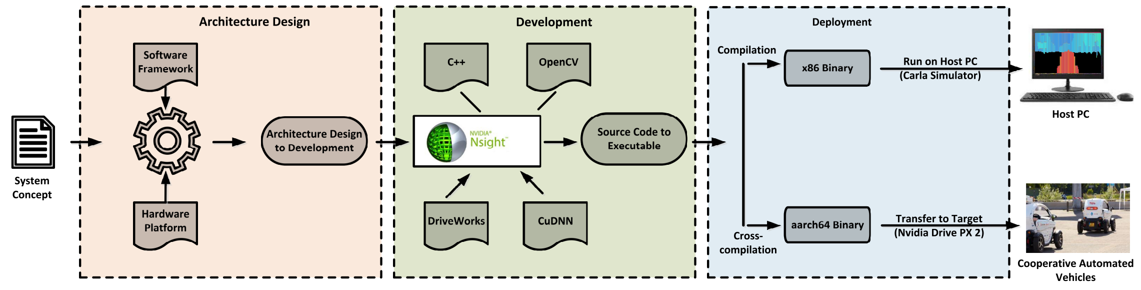

Fig. 4 A stixel-based stereo perception framework development process for cooperative automated vehicles.

4. Lane Detection

The lane detection sub-module is used to provide lane markings information in front of the ego vehicle. It uses Nvidia's proprietary DNN called *LaneNet* [44] to perform real-time lane detection and classification. The input image to this network is the RGBA image format, and the output is polylines representing lane markings. It calculates a probability map of lane markings for each pixel using an encoder-decoder architecture on the input image. The probability map is then binarized into clusters of lane-markings through polylines fitted to assign lane position types. The output of this algorithm is the lane markings represented as polylines and labels for the identified lane lines. It detects and classifies four different types of lane markings. The classified lane markings are left adjacent-lane, left ego-lane, right ego-lane, and right adjacent-lane when they are present on the road. The colors of the polylines represent the lane marking position types are as follows: yellow for left adjacent-lane, red for left ego-lane, green for right ego-lane, and blue for the right adjacent-lane.

5. Freespace Detection

The freespace detection sub-module is used to estimate the drivable freespace in front of the ego vehicle. It uses Nvidia's proprietary DNN called *FreeSpaceNet* [44] to perform the real-time drivable freespace boundary detection and classification. The input to the network is the RGBA image format, and the output is the freespace boundary for the recognized drivable freespace. The boundary separates the obstacle from the drivable freespace. Each pixel on the boundary is associated with one of the four semantic labels: red for the vehicle, blue for a pedestrian, green for the curb, and yellow for others.

We configured Nsight Eclipse Edition as Integrated Development Environment (IDE) to enable the entire host- and cross-compilation process. The above modules are developed in the Nsight Eclipse IDE using C++ with Nvidia DriveWorks 1.2, OpenCV 3.4.0, CUDA 9.2, and CuDNN 7.4.1 on an Ubuntu 16.04 LTS. The outline of our development process is shown in Figure 4.

D. Deployment

Initially, we ensure the source code compiles and runs (x86 binary) correctly on Host PC (on an x86 Desktop PC with an NVIDIA GPU) with Carla simulation before deploying on Drive PX2. Later, we transferred the cross-compiled version of the code (aarch64 binary) to the Drive PX2 and executed it in real time. Figure 4 shows the compilation and execution flow.

IV. Experimental Setup and Test Results

In this section, we describe the experimental setup, test results, and evaluate the performance of the proposed perception framework.

A. Experimental Setup

Experimental tests have been carried out with a Renault Twizy cooperative automated vehicle research platform, consisting of one lead vehicle and one ego vehicle. The ego vehicle of platooning equipped with a GMSL stereo camera (fixed on the car roof using a rigid mounting bar), and Drive PX2 hardware platform (fixed on the car roof using a mounting box) are shown in Figure 5. As an input sensor, we use a custom-built stereo camera using two 2.3 MP

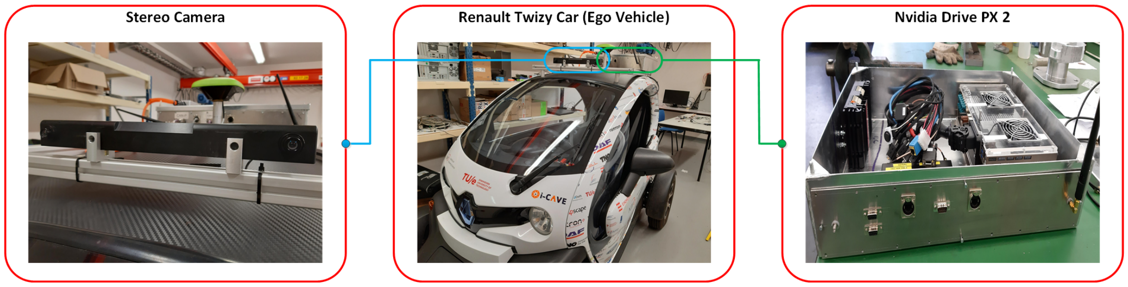

Fig. 5 A Renault Twizy cooperative automated vehicle research platform equipped with Drive PX2, and GMSL Stereo Camera. Stereo camera and Drive PX2 are fixed in an ego vehicle (center image). Custom-built stereo camera is mounted on the car roof using a rigid mounting bar (left image). Drive PX2 is placed on the roof of the ego-vehicle using a mounting box (right image).

Sekonix GMSL cameras with an ONSEMI CMOS AR0231 image sensor, an automotive-grade and high-resolution camera that is compatible with an automotive-grade hardware platform. The stereo camera consumes raw video or live camera input data from a calibrated AR0231 (revision >= 4) sensor with an RCCB color filter and a resolution of 1920 x 1208 pixels at 30 FPS (frames per second). It then reads the rig configuration JSON file, which contains the calibration of the camera (intrinsics and extrinsics) and performs stereo rectification. It then modifies the images by cropping and down-scaling by half. After these modifications, our framework provides disparity images at a resolution of 960 x 604 pixels. We process the disparity image using the multi-layered stixel-world algorithm [8] and includes the extensions as presented for the disparity-baseline system discussed in [23], which was developed for automotive applications. To compute the stixels, we use a Drive PX2 AutoChauffeur Tegra A System on Chip (SoC) that features six 64-bit CPU ARM cores (four ARMv8 Cortex A57 cores, two ARMv8 Denver 2.0 cores), 1152 CUDA cores (discrete GPU), and 256 CUDA cores (integrated GPU). We use an in-vehicle 12V DC USB powered HDMI 10.1 inch, small computer monitor to display stixels results to the user by streaming video, including the obstacle information computed by the stixel algorithm.

B. Experimental Results

The stereo perception framework has been extended with the stixel-world implementation. The framework is built on the Nvidia DriveWorks 1.2 SDK. It is tested on a moderately powerful PC with AMD Ryzen 1700X CPU, GTX 1060 6GB GPU, and 32GB of RAM. It runs real-time at an average of 30 FPS with stixel visualization enabled and at around 40 FPS with stixel visualization disabled (stixel computation is running). It consumes around 1.2GB system memory and 1.06GB graphics memory, with 15% CPU utilization and 71% GPU utilization. Figure 6 shows the visualization of all the modules in the stixels based stereo perception framework.

C. Performance Evaluation

We measured the Drive PX2 platform's computational performance and system resource utilization when running the stixels code. As a baseline, we also measure the same algorithm running a moderately powerful PC equipped with an AMD Ryzen 1700X CPU, 32GB RAM, and a GTX 1060 6GB GPU. To measure the performance, we processed the disparity images, and we present the average processing time of each module, which is shown in Table 1, the system resource utilization is shown in Table 2, the system power consumption is shown in Table 3.

V. Adaptation of Stixel-Based Stereo Perception from Cooperative Automated Vehicles to Multi-Robot Systems

We can see many similarities between cooperative automated vehicles and multi-robot systems. The framework can be further used for multi-robot systems consisting of UAVs, AGVs, ASRs, etc. Here we consider a use case with two UAVs, a lead UAV, and a follower UAV. The pilot manually controls the lead UAV, and the follower UAV autonomously follows the lead UAV. The proposed framework is deployed on the follower UAV, which continuously perceives the environment from an onboard front-facing stereo camera. It processes the perceived environment information and updates the autopilot of the follower UAV in real time. The autopilot of the follower UAV computes a control command

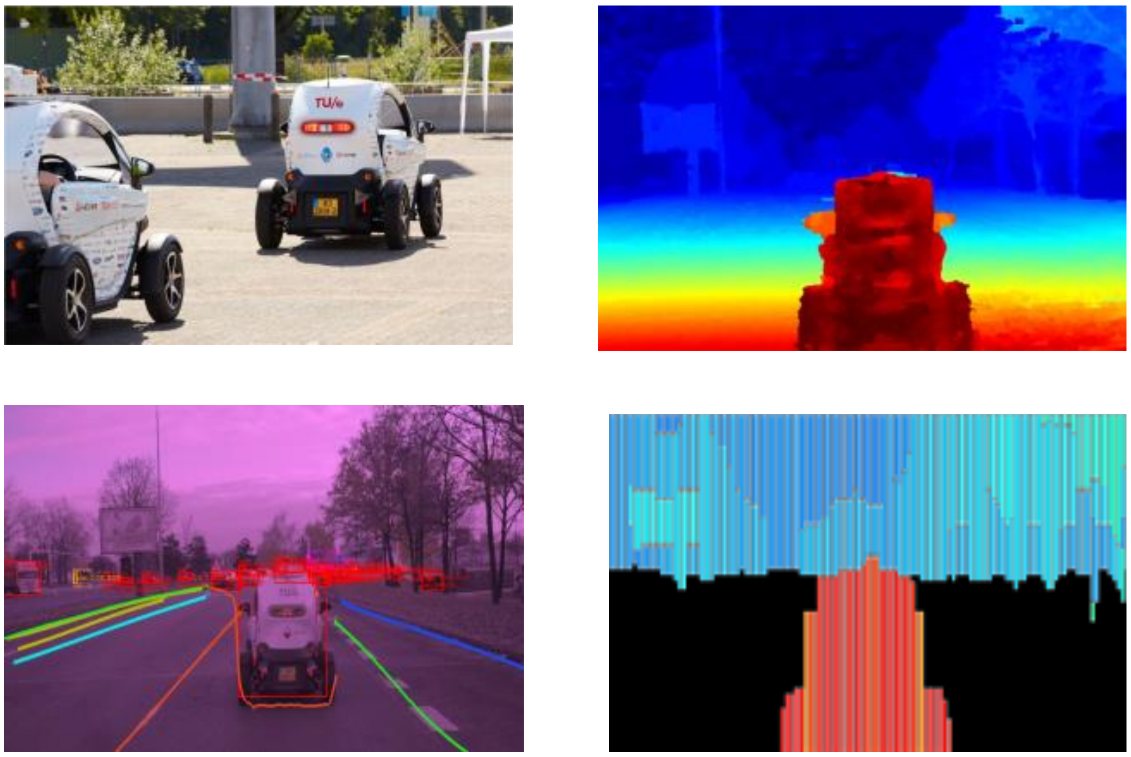

Fig. 6 A stixel-based stereo perception framework for cooperative automated vehicles: (a) Top left image: Renault Twizy used for platooning (Lead-follower vehicle). (b) Top right image: Stereo disparity map as the input image (Stereo camera's view). (c) Bottom left image: Obstacle detection (Stereo camera's view). (d) Bottom right image: Real-time stixels output image (Stereo camera's view).

Table 1 Execution time of each module.

Modules	Execution time (ms)
Stereo Rectification	3.193
Stereo Disparity	6.324
Freespace Detection	1.318
Object Detection	6.775
Lane Detection	1.965
Radial Distance	0.353
Motion Estimation	0.255
Stixel Computation	3.432
Stixel Computation with Visualization	10.796

Table 2 System resource usage by the framework.

System Resource	Usage
CPU (AMD Ryzen 1700X)	15%
System Memory (RAM)	1.2 GB
GPU (Nvidia GTX 1060 6GB)	71%
GPU Memory	1.06 GB

Table 3 System power consumption by the framework.

System Power	Consumption
PC with GPU (Nvidia GTX 1060)	450 W
Drive PX2 (Tegra A)	300 W

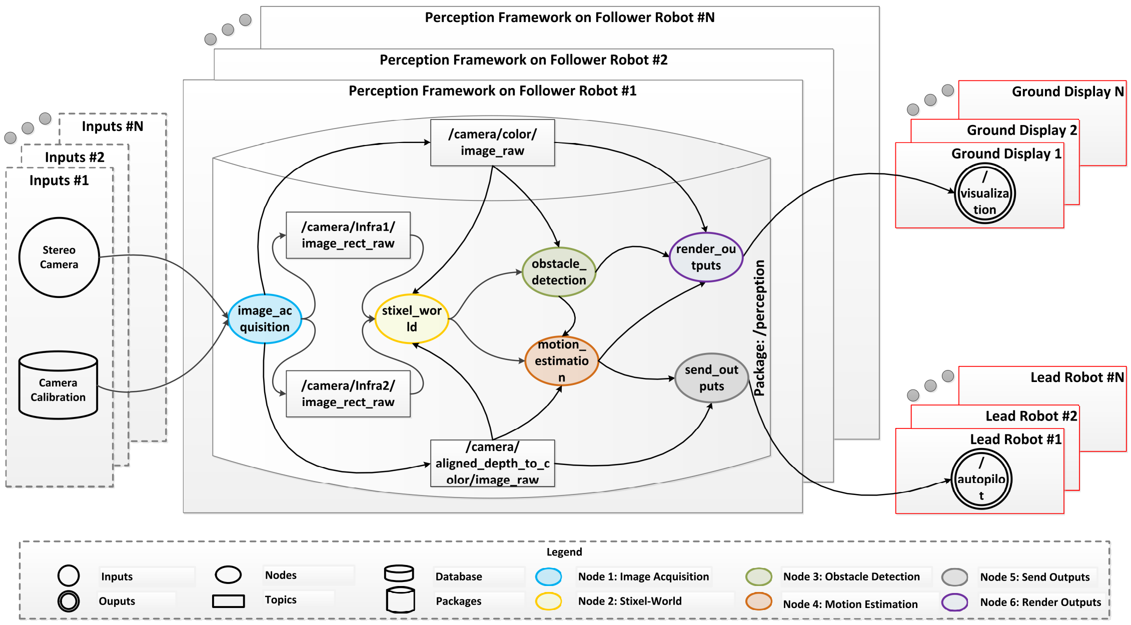

Fig. 7 A stixel-based stereo perception framework for multi-robot systems: (Node 1) Image acquisition, (Node 2) Stixel-world, (Node 3) Obstacle detection, (Node 4) Motion estimation, (Node 5) Send outputs, and (Node 6) Render outputs.

based on the processed environment information. It provides computed control commands to the autopilot in such a way that the follower UAV follows the lead UAV autonomously. Also, it visualizes the perceived environment information on the ground display system in real time. The outline of adapted perception framework for multi-robot systems is shown in Figure 7. It integrates the ROS middleware with the Ubuntu 18.04 ROS Melodic on the host PC and target platform, such as Nvidia Jetson Nano, Jetson Xavier, Jetson Orin, and Jetson TX2 hardware, a stereo camera such as Intel RealSense D435, Stereolabs ZED stereo camera, and it can realize all modules shown in Figure 2 and their functionality.

VI. Conclusion

This paper presents a stixel-based stereo perception framework for multi-robot systems. The framework is built on DriveWorks with CUDA programming. It is tested on a moderately powerful PC with AMD Ryzen 1700X CPU, GTX 1060 6GB GPU card, and 32GB RAM. It runs real-time at an average of 30 FPS with stixel visualization enabled and at around 40 FPS with stixel visualization disabled (stixel computation is running). We discussed the technical feasibility of adaptation of the developed framework for cooperative automated vehicles to multi-robot systems. Future work should deploy and test this framework with multi-robot systems.

Acknowledgments

We thank the generous support of the EPSRC Prosperity Partnership project (EP/V037846/1) on Swarm Spatial Sound Modulators.

References

[1] Kemsaram, N., Das, A., and Dubbelman, G., "A model-based design of an onboard stereo vision system: obstacle motion estimation for cooperative automated vehicles," *SN Applied Sciences*, Vol. 4, No. 7, 2022, p. 199.

[2] Queralta, J. P., Taipalmaa, J., Pullinen, B. C., Sarker, V. K., Gia, T. N., Tenhunen, H., Gabbouj, M., Raitoharju, J., and Westerlund, T., "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *Ieee Access*, Vol. 8, 2020, pp. 191617–191643.

[3] Gregory, J., Fink, J., Stump, E., Twigg, J., Rogers, J., Baran, D., Fung, N., and Young, S., "Application of multi-robot systems to disaster-relief scenarios with limited communication," *Field and Service Robotics: Results of the 10th International Conference*, Springer, 2016, pp. 639–653.

[4] Deka, A., and Sycara, K., "Natural emergence of heterogeneous strategies in artificially intelligent competitive teams," *Advances in Swarm Intelligence: 12th International Conference, ICSI 2021, Qingdao, China, July 17–21, 2021, Proceedings, Part I*, Springer, 2021, pp. 13–25.

[5] Leitner, J., "Multi-robot cooperation in space: A survey," *2009 Advanced Technologies for Enhanced Quality of Life*, 2009, pp. 144–151.

[6] Badino, H., Espinosa, A., Franke, U., and Pfeiffer, D., "The Stixel World - A Compact Representation of the 3D-World," *Pattern Recognition, 31st DAGM Symposium, Jena, Germany*, 2009, pp. 51–60.

[7] i-Cave participation ITS European Congress 2019, `https://i-cave.nl/i-cave-deelname-its-european-congress-2019/`, 2019. [Online].

[8] Pfeiffer, D., "The Stixel World–A Compact Medium-level Representation for Efficiently Modeling Dynamic Three-dimensional Environments," 2011.

[9] Kemsaram, N., Das, A., and Dubbelman, G., "A Stereo Perception Framework for Autonomous Vehicles," *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, 2020, pp. 1–6.

[10] Kemsaram, N., Das, A., and Dubbelman, G., "Architecture Design and Development of an On-board Stereo Vision System for Cooperative Automated Vehicles," *Proceedings of 23rd IEEE International Conference on Intelligent Transportation Systems 2020 (IEEE ITSC 2020), Rhodes, Greece*, 2020.

[11] Scharstein, D., and Szeliski, R., "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International journal of computer vision*, Vol. 47, No. 1-3, 2002, pp. 7–42.

[12] Hirschmuller, H., and Scharstein, D., "Evaluation of cost functions for stereo matching," *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2007, pp. 1–8.

[13] Gong, M., Yang, R., Wang, L., and Gong, M., "A performance study on different cost aggregation approaches used in real-time stereo matching," *International Journal of Computer Vision*, Vol. 75, No. 2, 2007, pp. 283–296.

[14] Hirschmüller, H., Innocent, P. R., and Garibaldi, J., "Real-time correlation-based stereo vision with reduced border errors," *International Journal of Computer Vision*, Vol. 47, No. 1-3, 2002, pp. 229–246.

[15] Hirschmuller, H., "Accurate and efficient stereo processing by semi-global matching and mutual information," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2, IEEE, 2005, pp. 807–814.

[16] Konolige, K., "Small vision systems: Hardware and implementation," *Robotics research*, Springer, 1998, pp. 203–212.

[17] Gehrig, S. K., Eberli, F., and Meyer, T., "A real-time low-power stereo vision engine using semi-global matching," *International Conference on Computer Vision Systems*, Springer, 2009, pp. 134–143.

[18] Hands-on Drive, `https://developer.nvidia.com/drive/`, 2020. [Online].

[19] Pfeiffer, D., and Franke, U., "Towards a Global Optimal Multi-Layer Stixel Representation of Dense 3D Data." *BMVC*, 2011, pp. 1–12.

[20] Benenson, R., Timofte, R., and Gool, L. J. V., "Stixels Estimation Without Depth Map Computation," *IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain*, 2011, pp. 2010–2017.

[21] Hernandez-Juarez, D., Espinosa, A., Moure, J. C., Vázquez, D., and López, A. M., “GPU-Accelerated Real-Time Stixel Computation,” *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 1054–1062. https://doi.org/10.1109/WACV.2017.122.

[22] Pfeiffer, D., Gehrig, S., and Schneider, N., “Exploiting the power of stereo confidences,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 297–304.

[23] Sanberg, W. P., Dubbelman, G., and de With, P. H. N., “Extending the stixel world with online self-supervised color modeling for road-versus-obstacle segmentation,” *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2014, pp. 1400–1407.

[24] van de Wouw, D. W. J. M., Sanberg, W. P., Dubbelman, G., and de With, P. H. N., “Fast 3D Scene Alignment with Stereo Images using a Stixel-based 3D Model,” *Int. Conf. on Computer Vision Theory and Applications (VISAPP)*, 2018, pp. 250–259. https://doi.org/10.5220/0006535302500259.

[25] Schneider, L., Cordts, M., Rehfeld, T., Pfeiffer, D., Enzweiler, M., Franke, U., Pollefeys, M., and Roth, S., “Semantic stixels: Depth is not enough,” *2016 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2016, pp. 110–117.

[26] Enzweiler, M., Hummel, M., Pfeiffer, D., and Franke, U., “Efficient stixel-based object recognition,” *2012 IEEE Intelligent Vehicles Symposium*, IEEE, 2012, pp. 1066–1071.

[27] Sanberg, W. P., Dubbelman, G., and de With, P. H. N., “Free-Space Detection with Self-Supervised and Online Trained Fully Convolutional Networks,” *IS&T Electronic Imaging - Autonomous Vehicles and Machines (EI-AVM)*, 2017, pp. 54–61. https://doi.org/10.2352/ISSN.2470-1173.2017.19.AVM-021.

[28] Sanberg, W. P., Dubbelman, G., and de With, P. H. N., “ASTEROIDS: A Stixel Tracking Extrapolation-based Relevant Obstacle Impact Detection System,” *IEEE Trans. on Intelligent Vehicles (T-IV)*, 2020, p. (IN PRESS).

[29] Autoware Auto is the world’s first "All-in-One" open-source software for autonomous driving technology, https://www.autoware.auto/, 2020. [Online].

[30] Baidu Apollo Open Platform, https://apollo.auto/, 2020. [Online].

[31] Comma AI OpenPilot Platform, https://comma.ai/, 2020. [Online].

[32] NXP BlueBox: Autonomous Driving Development Platform, https://www.nxp.com/design/development-boards/automotive-development-platforms/nxp-bluebox-autonomous-driving-development-platform:BLBX/, 2020. [Online].

[33] Nvidia DriveWorks Software Development Kit, https://developer.nvidia.com/drive/driveworks/, 2020. [Online].

[34] MobileEye EyeQ5, https://www.mobileye.com/our-technology/evolution-eyeq-chip/, 2020. [Online].

[35] Google TPU v3, https://cloud.google.com/blog/products/gcp/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu/, 2020. [Online].

[36] Texas Instruments TDA, https://www.ti.com/processors/automotive-processors/tdax-adas-socs/overview.html/, 2020. [Online].

[37] Xilinx Automotive Zynq UltraScale+ MPSoC ZCU104, https://www.xilinx.com/products/silicon-devices/soc/xa-zynq-ultrascale-mpsoc.html/, 2020. [Online].

[38] Habana Labs, https://habana.ai/, 2020. [Online].

[39] Nvidia GPUs, https://www.nvidia.com/en-us/deep-learning-ai/products/solutions/, 2020. [Online].

[40] Kemsaram, N., Das, A., and Dubbelman, G., “A SysML-based Design and Development of Stereo Vision System with Pose and Velocity Estimation for Cooperative Automated Vehicles,” *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, 2021, pp. 1–8. https://doi.org/10.1109/ICECET52533.2021.9698678.

[41] Kemsaram, N., Thatiparti, V. R. K., Guntupalli, D. R., and Kuvvarapu, A., “Design and development of an on-board autonomous visual tracking system for unmanned aerial vehicles,” *Aviation*, Vol. 21, No. 3, 2017, pp. 83–91. https://doi.org/10.3846/16487788.2017.1378265, URL https://doi.org/10.3846/16487788.2017.1378265.

[42] Cui, P., and Yue, F., “Stereo vision-based autonomous navigation for lunar rovers,” *Aircraft Engineering and Aerospace Technology*, 2007.

[43] Hernandez-Juarez, D., Chacón, A., Espinosa, A., Vázquez, D., Moure, J. C., and López, A. M., “Embedded real-time stereo estimation via semi-global matching on the GPU,” *Procedia Computer Science*, Vol. 80, 2016, pp. 143–153.

[44] Nvidia DriveWorks Development Guide, https://developer.nvidia.com/driveworks-docs/, 2020. [Online].