Automation in Construction 112 (2020) 103068

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

An integrated UGV-UAV system for construction site data collection

Khashayar Asadi[a,*], Akshay Kalkunte Suresh[b], Alper Ender[b], Siddhesh Gotad[b], Suraj Maniyar[b], Smit Anand[b], Mojtaba Noghabaei[a], Kevin Han[a], Edgar Lobaton[b], Tianfu Wu[b]

[a] Dept. of Civil, Construction, and Environmental Engineering, North Carolina State University, Raleigh, NC, USA
[b] Dept. of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA

ARTICLE INFO

Keywords:
Multi-agent robotic system
UGV
UAV
Autonomous data collection
Real-time construction monitoring

ABSTRACT

There have been recent efforts to increase the degree of automation and frequency of data collection for construction applications using Unmanned Aerial/Ground Vehicles (UAV/UGV). However, the current practice of data collection is traditionally performed, which is manual, costly, time-consuming, and error-prone. Developing vision-based mobile robotic systems that are aware of its surrounding and capable of autonomous navigation are becoming essential to many construction applications, namely surveying, monitoring, and inspection. Nevertheless, the systems above suffer from a series of performance issues. One major problem is inefficient navigation in indoor and cluttered scenes with many obstacles and barriers, where some places are inaccessible by a UGV. To provide a solution to this problem, this paper designs a UAV-UGV team that integrates two custom-built mobile robots. The UGV autonomously navigates through space, leveraging its sensors. The UAV acts as an external eye for the UGV, observing the scene from a vantage point that is inaccessible to the UGV. The relative pose of the UAV is estimated continuously, which allows it to maintain a fixed location that is relative to the UGV. The key aspects for the development of this system that is capable of autonomous navigation are the localization of both UAV and UGV, mapping of the surrounding environment, and efficient path planning using multiple sensors. The proposed system is tested in an indoor and cluttered construction-like environment. The performance of the system demonstrates the feasibility of developing and deploying a robust and automated data collection system for construction applications in the near future.

1. Introduction

Construction industry is one of the major economic sectors in most countries, with 9%– 15% of total Gross Domestic Product (GDP) being allocated to their built environment [1]. Despite this economic importance, this industry is plugged with low productivity and inefficiencies. During the past few decades, the productivity rate in many sectors has been steadily increasing; however, this rate in the construction industry has barely increased, and it may have even decreased [2]. Automated systems and robotics are known as technologies that have the potential to revolutionize the construction industry by addressing the productivity challenges while improving quality [3].

In the past few years, on-site semi-automated and automated Unmanned Vehicles (UVs) have received significant attention for construction applications. They have been proposed for various activities including inspection and structural health monitoring [4, 5], floor cleaning [6], building components production [7-9], building components assembly [10-12], material handling [13], and construction surveying and monitoring [14-19]. According to the applications in the latter activity, implementation of UVs for automated data collection is limited to the outdoor environment. Typical autonomous UVs on job sites use either GPS [20–22] or BIM-driven map [17, 23, 24] for autonomous navigation. GPS technology is mainly suitable for outdoor applications. The BIM-based motion planning solutions are inefficient in a cluttered construction site with many temporary structures that are not present in BIM.

An autonomous UV system capable of data collection on construction sites needs to have the following capabilities: 1) ability to collect multiple types of sensory and visual data that are required for construction performance monitoring, 2) ability to process the collected data in real-time using a low computational complex platform and 3) capability to navigate efficiently and autonomously on a construction site [25]. The authors' previous work on an integrated mobile robotic system [15] presents vision-based UGV for autonomous data collection on construction sites. This system addresses the first two concerns for an efficient autonomous data collection system. However, it is

* Corresponding author.
E-mail address: kasadib@ncsu.edu (K. Asadi).

https://doi.org/10.1016/j.autcon.2019.103068
Received 3 June 2019; Received in revised form 22 December 2019; Accepted 31 December 2019

inefficient for autonomous navigation in a cluttered indoor scene where all the locations are not accessible by the UGV for data collection. Sites occluded with barriers or surfaces with different elevations are some examples of such scenes.

To address this issue, this study proposes a mobile robotic system that integrates two custom-built aerial and ground UVs. The complementary skills provided by each vehicle overcome the specific limitation of the other. Unmanned aerial vehicles (UAVs) offer a broad field of view and rapid coverage of search areas, which is ideal for mapping and monitoring tasks. However, they are constrained by their low payload (hundreds of grams) in relation to their size and short operational time (tens of minutes). UGVs, on the other hand, can carry substantial payloads and operate for extended periods. They offer high-resolution sensing, but with less field of view and lower coverage speed compared to UAVs. They are much more susceptible to obstacles, occlusions, and other sensing limitations. Therefore, coordinated operations between UGVs and UAVs can create highly beneficial synergies, such as multi-domain sensing and enhanced line of sight communications.

The objective of this paper is to present a collaborative and explorative approach using UGV and UAV. To achieve this goal, the given UGV-UAV system periodically visits a set of places of interest. These places have been pre-selected by the construction management team. During this mission, the UGV autonomously moves on the site and continuously scans the environment by its sensors. The relative pose between the two vehicles is estimated consistently, which enables the UAV to follow the UGV's path during its navigation. If the place of interest is not accessible by the UGV due to environmental constraints, the UGV sends the UAV to the desired location to scan the area of interest. The UAV does not process all data onboard; instead, the sensor's data are transmitted to the UGV's onboard computer through the network, where all the processing takes place. The UAV then returns to the UGV, and the heterogeneous team continues towards the next area of interest. It is necessary to mention that the places of interest for data collection are selected by the construction management team either directly through the laptop on the UGV, before starting the mission, or by commanding the UGV remotely using Secure Shell (SSH), which enables the operator to provide waypoints even during the mission.

To have an ideal and efficient system for indoor surveillance and monitoring, a custom-built indoor blimp is designed as the UAV for this study. Use of indoor blimp limits the application of the system to indoor environments. However, outdoor blimps which are more stable against air disturbance, have the potential to makes the system suitable for outdoor applications. The use of blimp instead of other types of UAVs (e.g., quadrotor, hexacopter, etc.) has the advantages of being safe in a cluttered indoor space and having lower cost, energy consumption, and noise.

The main contributions of this paper are 1) a comprehensive literature review on previous integrated UAV-UGV systems with different applications and 2) an integrated UAV-UGV system that can autonomously navigate and collect visual data for construction monitoring applications. This system addresses most of the limitations of past studies (to be further detailed in Section 2.3 UAV-UGV collaborative systems). The critical aspects of the system are 1) localization of both UGV and UAV, (2) capability of being contextually aware of the environment, (3) mapping of the surrounding environment, and (4) efficient path planning. Based on these aspects, the proposed system processes the context awareness (via semantic image segmentation), localization, mapping, and control planning in real-time, as illustrated in Fig. 1. To evaluate the performance of the proposed system, the system is implemented on an indoor cluttered construction-like environment (to be further detailed in Section 5 Experimental setup and results) for data collection purposes, demonstrating the feasibility of real-time performance for construction applications.

2. Background

The proposed system focuses on autonomous cooperation between UGV and UAV for navigation and indoor environment monitoring. Simultaneous Localization and Mapping (SLAM) and image segmentation algorithms are used for robot's localization, environment mapping, and scene understanding. Therefore, this section provides background information for visual SLAM and scene understanding. Additionally, it dissects into the previous UAV-UGV cooperative systems and discusses their application, sensors for both UGV and UAV, level of human intervention, and limitations as benchmarks for the current and future systems.

2.1. SLAM

In locations where GPS cannot receive signals (i.e., indoor environments), Simultaneous Localization and Mapping (SLAM) can be used to estimate UV's position and build a map of an unknown environment simultaneously. Light Detection and Ranging (LIDAR) based SLAM and vision-based SLAM are two major methodologies of SLAM algorithms. Each of these methodologies has its limitations and advantages. For instance, the vision-based SLAM, specially monocular SLAM, has higher computational cost compared to LIDAR-based SLAM approaches [26]. However, with recent advances in computer vision and processing power, visual SLAM can now run in real-time. On the other hand, LIDAR-based methods have high sensor weight and energy consumption. They are mostly inefficient in outdoor scenarios. In the proposed system, visual SLAM is implemented for both UGV and UAV and Robot Operating System (ROS) is used for data exchange between various modules. Therefore, this subsection provides some background information about the latest visual SLAM approaches that are compatible with ROS.

Visual SLAM techniques estimate the camera's location and orientation (known as camera pose) of image sequences and map the environment, simultaneously in real-time [19]. Visual SLAM compares visual features to find similarities between consecutive image frames to estimate the camera poses (relative to the previous frame). It also uses the corresponding features of the consecutive images to map the environment as it runs. The overall goal of this section is to review and discuss the state-of-the-art visual SLAM algorithms which have the potential to be implemented in the proposed system. Then, the most appropriate SLAM methods for implementation on each of the UGV and UAV are selected. Finally, the reason behind each selection is explained.

Although many open-source visual SLAM approaches exist, not many are compatible with robotic systems through ROS [27]. The authors' previous work used a monocular camera as a visual sensor and ORB-SLAM as the SLAM algorithm for localization purposes. ORB-SLAM has a scale issue. To avoid dealing with the scale ambiguity, this section only discusses methods that enable real scale estimation while mapping (e.g., using a stereo camera or visual inertial odometry). Therefore, monocular approaches like Semi-direct Visual Odometry (SVO) [28], Deferred Triangulation SLAM (DT-SLAM) [29], Large-Scale Direct monocular SLAM (LSD-SLAM) [30], and Oriented FAST and rotated BRIEF SLAM (ORB)-SLAM [31] are excluded in this section. The following visual SLAM methods are applicable to ROS and do not suffer from scale drift over time.

Stereo Parallel Tracking and Mapping (S-PTAM) [32] and ORB-SLAM2 [33] are feature-based approaches, that mostly are used with a stereo camera. In these approaches, Distributed Bag of Words (DBoW2) [34] is used for loop closure detection, which optimizes the map through bundle adjustment. After loop closure, the graph optimization runs in a separate thread. Therefore, the camera tracking frame rate performance is not affected. As the map grows, the processing time for loop closure detection and graph optimization is increased, which

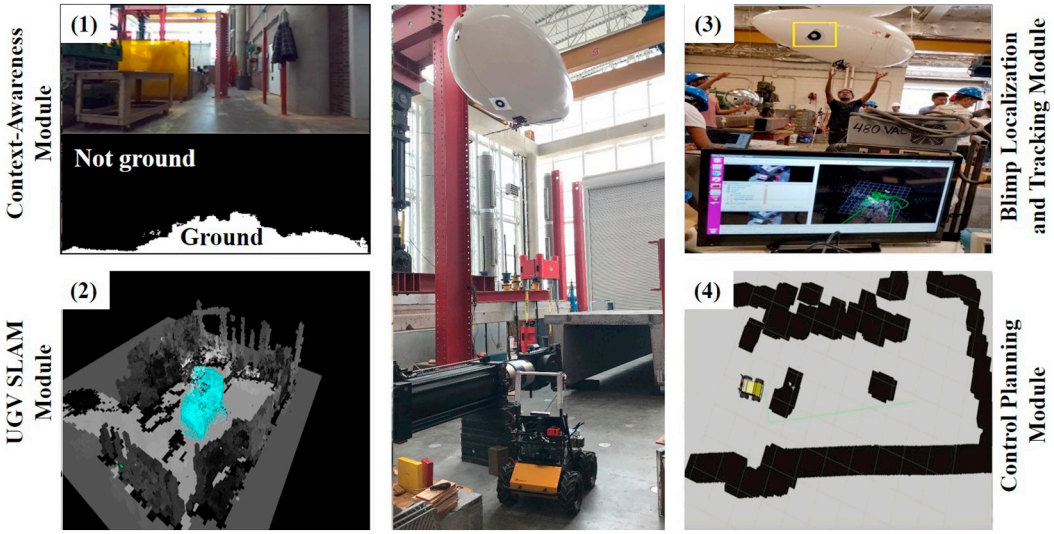

Fig. 1. Overview of the system with different modules: (1) Context Awareness Module, (2) UGV SLAM Module, (3) Blimp Localization and Tracking Module, and (4) Control Planning Module.

can lead to a significant delay for loop closure correction after the loop is detected. The main drawback with these methods is that the output point cloud is a sparse point cloud and can not be used for occupancy map generation suitable for autonomous navigation.

Red Green Blue inverse Depth (RGBiD) SLAM [35] and Dense Visual Odometry (DVO)-SLAM [36] use photometric and depth errors for pixels of RGB-D images to estimate motions, instead of local visual features. Although they provide dense point clouds of the environment, they lack a loop closure detection approach, which makes them inefficient for large-scale mapping.

Monocular Visual-Inertial System (VINS-Mono) [37] and maplab [38] are visual-inertial graph-based SLAM systems consisting of a monocular camera and an low-cost Inertial Measurement Unit (IMU). Using this minimum sensor suite, they provide a real-scale estimate of camera poses, as well as sparse representation of the environment in terms of a point cloud. The lightweight sensor system makes them the most suitable SLAM approaches for autonomous systems with a limited payload.

One of the assumptions with the above approaches is that camera frames always have enough visual features to track. However, in construction environments, a camera can be fully obstructed from different objects or face feature-less surfaces (e.g., white wall) during navigation. Red Green Blue Depth Simultaneous Localization and Mapping (RGB-DSLAMv2) [39] and Real-Time Appearance-Based Mapping (RTAB-MAP) [40] use external odometry as motion estimation to be more robust to such events. They both can additionally create a 3D occupancy grid named as OctoMap [41] (RTAB-Map can also generate 2D map) and a dense point cloud of the environment. Besides these two methods, none of the above visual SLAM approaches create the required occupancy grid map for autonomous navigation. It should be noted that both approaches are compatible with Kinect as a sensor. RTAB-Map is also compatible with a stereo camera.

In this paper, localizing the UVs and estimating their relative positions are necessary for motion planning. Therefore, VINS-Mono is implemented on the UAV to provide real-scale odometry. This method is known as suitable SLAM approach for UAVs with a limited payload. On the other side, RTAB-Map is implemented on the UGV to provide real-scale odometry together with dense point clouds suitable for occupancy map generation. This SLAM algorithm is compatible with a stereo camera and has been proved as proper SLAM approach for large-scale, long-term, and real-time operations [42].

2.2. Semantic segmentation

In robotics research, object classification and semantic segmentation are well-known approaches for scene understanding, which enable a robot to recognize objects of interest and make a proper decision based on its observation. Object recognition uses bounding boxes to classify objects in the scene, whereas semantic segmentation provides each pixel with a label, which results in a boundary around the objects. While semantic segmentation has more computational cost for real-time application, it would provide more accurate boundaries for the objects. The accurate boundary provides the robot with a classified scene that is used for object avoidance and autonomous navigation.

In this paper, the real-time performance of the proposed system is one of the priorities. Therefore, efficient Convolutional Neural Networks (CNNs) for mobile robotic applications with a light model is required for the system. Semantic segmentation networks such as MobileNets [43], ENet [44], and LNSNet [45, 46]) are examples of such efficient models that are designed to run on robotic systems with limited computing resources.

In the authors' previous study [15], ENet was implemented as the semantic segmentation model. However, this segmentation task had still the heaviest computational load on the system. The large segmentation model size and the high memory usage restricted the authors to run the segmentation module on a separate processor. Data transformation between segmentation and SLAM modules over the network caused a high latency (inference speed of 15 frames per seconds (fps) before integration is reduced to 3 fps after integration). Therefore, LNSNet [45] with smaller model size and faster inference speed compared to the ENet model was proposed by authors to determine navigable space in real-time on construction sites. In this paper, authors implement LNSNet as the semantic segmentation method, that enables multiple modules to be combined and run on the same processor unit that also runs other modules.

2.3. UAV-UGV collaborative systems

Cooperative robotics has been explored since the 1990s when challenges with collaboration between unmanned vehicles (UVs) and

Table 1
Related studies in multi-agent systems consisting of UAV and UGV.

References	UGV Platfotm	UAV Platform	UGV Sensors							UAV Sensors							Additional Remarks	OCU
			Monocular Camera	Stereo Camera	Kinect	IMU	Laser Scanner	GPS	Ultra Sonic	Monocular Camera	Stereo Camera	Kinect	IMU	Laser Scanner	GPS	Ultra Sonic		
UAV-UGV Relative Positions are Estimated																		
Downs et al. (2003) [49]	N/S	N/S					■	■						■	■		An overlap between the aerial and ground maps is required.	
Forster et al. (2013) [50]	Kuka YouBot	NanoQuad MAV			■					■			■				An overlap between the aerial and ground maps is required.	
UGV is Followed by UAV or Vice Versa																		
Tucker et al. (2010) [53]	N/S	Blimp						■							■		Has not been tested in realistic environment.	■
Cantelli et al. (2013) [52]	ETNAMATICA S.r.1	Quadrotor	■			■	■	■		■							A WebGIS platform is required. The UGV is controlled manually.	■
Hood et al. (2017) [51]	Kobuki TurtleBot 2	Quadrotor	■				■			■							Random movement is considered for the UGV. UAV is limited to be in the close proximity of the UGV.	■
Targets are Searched and Tracked																		
Grocholsky et al. (2006) [55, 56]	4WD model trucks	Fixed-wing aircrafts		■		■		■		■					■		Is efficient in free of occlusions environments with fixed targets	■
Moseley et al. (2009) [76]	iRobot PackBot	Fixed-wing aircraft	■	■			■	■		■							A human operator is required to provide waypoints.	■
Owen et al. (2010) [54]	Pololu 3π line-following	Quadrotor															8 cameras are installed in the site.	■
Hui et al. (2013) [58]	3-wheeled omni-directional	Quadrotor								■			■				UGV is controlled manually.	■
Ghamry et al. (2016) [57]	Two-wheel Quanser	Quadrotor															24 OptiTrack cameras are installed in the site.	■
UGV's Navigation Performance is Improved Uing a Map Provided by UAV																		
Chaimowicz et al. (2005) [69]	5 model trucks	Fixed-wing aircraft & blimp		■		■		■		■			■		■		A human operator is required to provide waypoints.	■
Luo et al. (2011) [77]	2 micro UGVs	Quadrotor	■						■	■							UAV flies in a predetermined path	■
Luo et al. (2011) [65]	2 micro UGVs & LEGO NXT	Quadrotor	■				■		■	■							UGV is controlled manually. UAV flies in a predetermined path	■
Giakoumidis et al. (2012) [66]	2-wheel Pioneer P3-DX	Quadrotor								■						■	6 cameras are installed in the site. UAV flies in a predetermined path	■
Garzon et al. (2012 & 2013) [59, 78]	Pioneer3-AT	Quadrotor	■			■		■		■			■		■	■	UAV is limited to be in the close proximity of the UGV. UAV is controlled manually.	■
Al-Jarrah and Roth (2014) [60]	N/S	Blimp								■							No navigation. UAV is limited to be in the close proximity of the UGV.	■
Kim et al. (2014) [61]	N/S	2 quadrotors								■							No navigation. UAV is limited to be in the close proximity of the UGV.	■
Mueggler et al. (2014) [67]	Kuka YouBot	Quadrotor					■			■			■				UAV flies in a predetermined path	■
Harik et al. (2015) [62]	Unicycle-like, wheeled robot	Quadrotor	■							■							UAV is controlled manually. A human operator is required to provide waypoints.	■
Fankhauser et al. (2016) [63]	Walking robot	Hexacopter	■			■	■			■							UAV is controlled manually.	■
Christie et al. (2017) [68]	Turtle	Helicopter					■	■		■				■	■		UAV flies in a predetermined path	
Kandath et al. (2018) [64]	Custom-built robot	Quadrotor						■		■					■		UAV is controlled manually.	
Peterson et al. (2018) [70]	Clearpath Jackal	Custom-built hexacopter				■	■	■		■			■		■		UAV is controlled by an operator using waypoints. Obstacles are denoted with a known, solid color.	■
Collaborative Surveillance While Mapping the Environment																		
Saska et al. (2012) [72]	Pioneer3-AT	Quadrotor	■							■							UGV moves in a predetermined path. UAV searches for ground markers for navigation.	
Michael et al. (2012) [71]	Kenaf and Quince	Quadrotor	■				■					■	■	■			UGV is controlled manually. A human operator is required to provide waypoints for the UAV.	■
Batzdorfer et al. (2017) [74]	2 Robotnik Summit XL	3 hexacopters	■	■		■	■	■		■			■		■		A priori known map is required. A human operator is required to provide waypoints.	■
Qin et al. (2019) [73]	Custom-built robot	Quadrotor			■		■				■		■	■			UAV is controlled manually.	■
Proposed system	Custom-built robot	Custom-built blimp	■	■			■			■			■					

possible future applications were discussed by Cao et al. [47]. In terms of perception abilities, payload, and mobility, UGVs and UAVs have several advantages and disadvantages over each other. Over the past 15 years, studies have focused on creating multi-robot systems to combine UGV's and UAV's advantages for applications in many fields, such as military and surveillance [48]. Based on past studies, efforts in this area are mainly categorized into the five following groups: (1) UAV-UGV relative positions are estimated, (2) UGV is followed by UAV or vice versa, (3) Targets are searched and Tracked, (4) UGV's Navigation performance is improved using a map provided by UAV, and (5) collaborative surveillance while mapping the environment. Table 1 shows the studies related to each category. In this table, the UAV and UGV platforms alongside with the sensors used in each platform are mentioned. The specific requirements for each system are provided as additional remarks in the table. As shown in the table, the Operator Control Unit (OCU) is required for most of the studies, which is capable of providing coordination between the operator and the robots. Data from different agents are sent to this unit for further processes. The results are sent back to the agents as different commands (e.g., waypoints provided by the operator for autonomous navigation).

Systems in the first category focus on estimating the relative position of mobile agents using registration of the map generated by each agent [49, 50]. Autonomous navigation is not considered for vehicles in any of the studies in this category. For continuous localization, an overlap between the maps that are generated by each robot is necessary, which restricts the robots to capture data from almost similar scenes.

In the second category, a UGV is tracked using a sensor mounted on the UAV [51] or vice versa [52]. The tracked agent can be localized only if it is in the field-of-view of the other agent. Therefore, they are required to stay close to each other at all times. In surveillance scenarios, this constraint prevents one agent from exploring areas further away from the other one, which makes the global guidance mission inefficient. Furthermore, navigation for the tracked agent happens either manually by the operator [52] or by random-based movements [51]. In some studies, it even has not been specified [53], and the system never been tested in a realistic environment.

Studies related to targets searching and tracking include efforts on searching for fixed targets in an environment using cooperative air and ground surveillance, tracking a moving target using a UAV in cooperation with a UGV [54–56,76], and strategies for take-off, tracking and landing of a UAV on a UGV [57, 58]. Autonomous navigation for either UAV or UGV has not been considered in the above studies, and they all rely on either manual navigation or waypoints provided by OCU operator. In systems proposed in Refs. [54, 57] , no sensor is used on the UGV and UAV platforms. Instead, multiple cameras are set up in the test environment. They provide robots with information related to localization and navigation.

In the fourth category, UAV is used as a remote sensor which flies ahead the UGV to provide the traversable map of the environment to the UGV for autonomous navigation. Similar to the above categories, most of the systems in this category rely on OCU for manually controlling the UAV, providing agents with waypoints for navigation, or preparing the map provided by the UAV for UGV navigation. Autonomous navigation for the aerial vehicle has not been considered in this category in the majority of cases. It is either controlled manually [59–64,78] or navigates in a predetermined path [65–68,77], or rely on waypoint-based navigation provided by the operator [69, 70].

Integrated robotic systems from the last category are the most similar systems to the current study. Such systems capable of collaborative surveillance and mapping the environment handle multiple tasks from the previous categories. For instance, in the proposed system during collaborative surveillance and mapping the environment, the relative position of the UAV with respect to the UGV is estimated (first category), and UAV follows the UGV during its navigation (second and third categories). Variety of works have been done for collaborative surveillance; however, relatively few studies operate both vehicles simultaneously and in an autonomous manner. The UGV in Ref. [71] is controlled manually by the operator. In the system proposed by Saska et al. [72], UGV moves in a predetermined path. Most of the efforts in this category rely on OCU for online data processing [73] or providing waypoints for the robots' navigation [71, 74].

2.4. Proposed system

The proposed system consists of two custom-built robots (one ground robot and one aerial blimp). A stereo camera and LIDAR are used on the UGV for localization, autonomous navigation, and environment mapping. A wide-angle camera is mounted on top of the UGV for blimp tracking and localization. The blimp uses a monocular camera in integration with an IMU sensor for localization purposes.

This system addresses most of the limitations of past studies. To clarify the advantages of the proposed system over the past studies, major limitations with the existing systems are pointed in Table 1 in the “Additional Remarks” column. These constraints are addressed in the proposed system. According to the Table 1, in majority number of past studies, the manual control is required for at least one the UVs; however, in the proposed system, both aerial and ground vehicles navigate autonomously. Lack of real-time data processing on the vehicles' onboard computers is one of the main research gaps in most of the existing systems, which has been addressed properly in the proposed system. In contrast to many of the existing systems, in the proposed system, there is no need for installing additional sensors (e.g., multiple OptiTrack cameras) in the site for localization purposes. Moreover, contrary to many of the studies presented in Table 1, the proposed system does not require high-level user interaction, such as OCU.

This paper is one of the few studies on using an indoor blimp in a realistic environment. To the best of our knowledge, there is no work that has been published on autonomous data collection using heterogeneous blimp and UGV in GPS-denied cluttered environments, especially for construction applications. The blimp robot not only provides an excellent opportunity for indoor environment monitoring but also increases the monitoring efficiency. The use of an indoor blimp instead of different types of UAVs used in past studies (e.g., drone, fixed-wing aircraft, etc.) has the following advantages 1) reduces safety concerns: in terms of hitting and falling, blimps are reasonably safe, 2) reduces cost: indoor blimps are significantly less expensive than drones, 3) reduces energy consumption and increases operation time: the blimp's battery consumption is significantly lower than a simple drone. Moreover, it can stay floated without the uplift force from a motor, which increases the flight time significantly, and 4) reduces noise: the blimp operation produces negligible noise. These advantages make the blimps ideal for indoor surveillance and monitoring without disturbing the environment [75].

3. Hardware description

The proposed integrated system consists of two custom-built unmanned and autonomous platforms, a ground vehicle and a blimp. Mounted in the center of the UGV's chassis is a laptop with the following specification to carry out most of the necessary calculations of all modules: 16 GB DDR3 RAM, Intel Core i7-4710HQ quad-core Haswell processor, and NVIDIA GeForce GTX 960M. Two Raspberry Pis, one on the UGV and the other on the UAV, are used to control the actuators. The laptop and Raspberry Pis are all connected to the same network via a router on the UGV and run both Ubuntu 16.04 LTS and

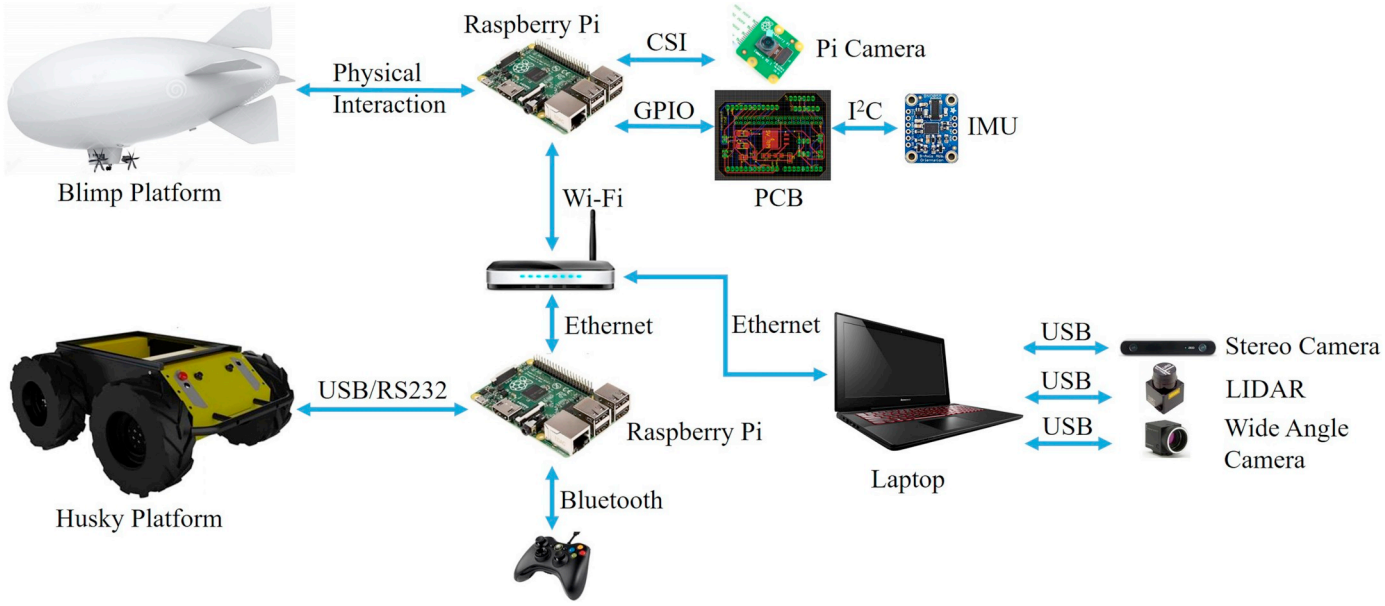

Fig. 2. Physical diagram of components in each vehicle. The channels used for interactions between the different components are labeled in blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

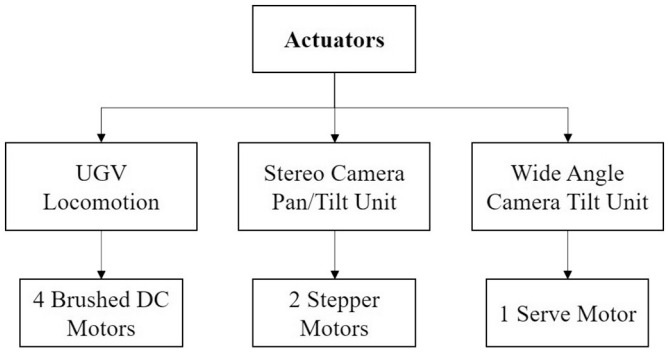

Fig. 3. UGV actuators diagram.

ROS Kinetic. The laptop and Raspberry Pi on the UGV are connected to this router using an Ethernet cable. The Raspberry on the blimp is then connected wirelessly via Wi-Fi to the router.

All modules run as individual nodes in the ROS ecosystem and use ROS Topics and Messages to exchange information. Fig. 2 illustrates the hardware utilized in each vehicle, including the sensors interfacing with different modules and the network diagram necessary for data transformation among the modules in the proposed system.

3.1. UGV

In the proposed system, UGV is a custom-built robot, that is built on a Clearpath Husky A200 mobile robotic platform [79]. On this platform, multiple actuators are used in locomotion and camera's pan/tilt unit (see Fig. 3). A Microsoft Xbox controller is used for manual navigation during the initial mapping of an unknown environment. The Raspberry Pi on the UGV receives the control commands from the Control Planning Module through Transmission Control Protocol/Internet Protocol (TCP/IP) and sends back the wheel encoder information. It can also operate a kill switch that remotely stops the motors on the UGV in case of an emergency.

The UGV is powered by a 24 V, 20 Ah battery. The operation time varies from 3 h in typical operation up to 8 h in standby (i.e., no motion) status [80]. The battery is used by the UGV for locomotion as well as to power multiple units, such as a router, Raspberry Pi, and laptop. Power from the battery is distributed at 20 V, 12 V, and 5 V for other units (see Fig. 4). The voltage conversion is happening using the buck (step-down) converters. Buck converters are more power-efficient than typical voltage regulators. Voltage regulators mostly step-down the voltage by dissipating power as heat, whereas, buck converters are capable of stepping down the voltage by storing the power in an inductor.

In addition to the above components, multiple sensors are used on the UGV. They are connected to the laptop through the USB connection. A wide-angle camera, Point Grey Flea3, is mounted on a tilt unit for blimp localization and tracking. As the blimp moves, this camera tracks the marker attached to the blimp enabling the system to continuously estimate the location of the blimp with respect to the UGV (to be further detailed in Section 4.3.1 Blimp localization using a marker). A HOKUYO URG 04LX-UG01 LIDAR sensor is mounted in the front of the UGV to generate a 2D map describing the occupancy of the environment. Moreover, to make the system contextually more aware of the environment, a forward-looking stereo camera (ZED) [81] is mounted on a pan-tilt unit to be used by Context-Awareness Module in integration with SLAM module in providing a 3D segmented map of the environment. The UGV's main components, including sensors and their functionality are highlighted in Fig. 5.

3.2. UAV

The UAV in this paper is a custom-built blimp that is made of 3D-printed parts and selected off-the-shelf components including a Raspberry Pi, a custom-built Printed Circuit Board (PCB), two motor controllers, three motors, four batteries, and multiple sensors. The blimp acts as a “flying external eye” for the UGV, observing a scene from a vantage point inaccessible by the UGV. The first step in the development of an aerial blimp is to design a flying mechanism consisting of an envelope and a payload carrier. To achieve this goal, the weight of the payload, including the hardware components, is estimated. Then, the size of the envelope is estimated. Correct estimation of the envelope's dimensions is necessary. If the envelope is bigger than

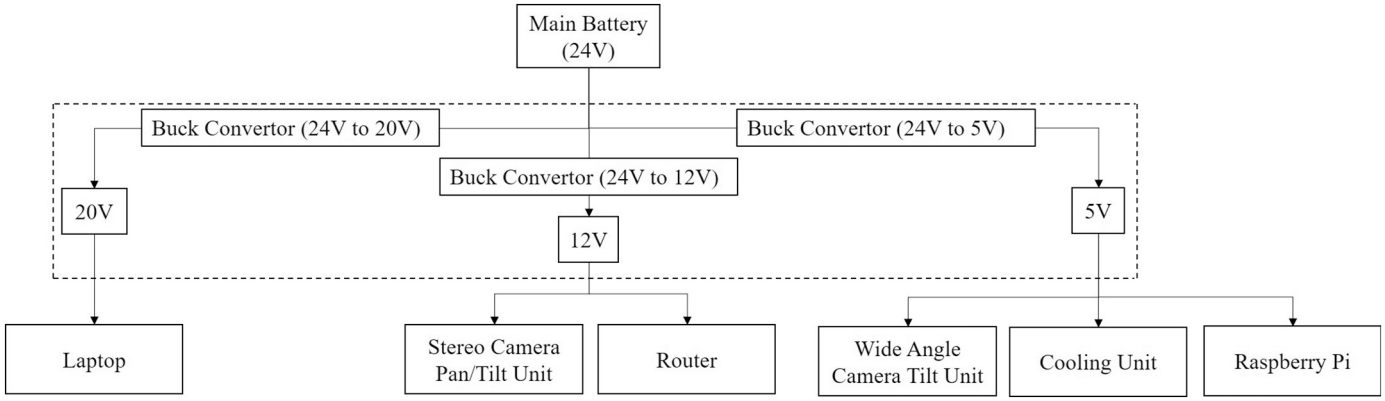

Fig. 4. Main battery power distribution diagram.

the required size, navigating in an indoor construction site is challenging for the blimp, and it will not be stable in an encounter with airflow (e.g., outdoor scenarios, indoor environments with active air conditioning system, etc.). If the envelope is smaller than the required size, it would not be able to lift the payload carrier. Many variables need to be considered, such as helium purity, payload carrier weight, and the shape and material of the envelope itself. The envelope used in the proposed system, is ellipsoid in shape, with a length of 1.83 m, made of formulated polyurethane (vinyl) that is chosen for its excellent helium holding properties, flexibility, and lightweight. The blimp can carry a total weight of 300 g.

The 3D CAD model of the payload carrier is designed, and 3D printed. It is attached at the base of the ellipsoid envelope and is designed to carry the hardware components of the flying blimp and the motor control system. The blimp has a single board computer (Raspberry Pi 3), to which all the sensors and actuators are interfaced. The in-built Wi-Fi module, available ports for micro SD card, GPIO, and camera together with its lightweight, makes Raspberry Pi 3 suitable for the proposed system.

The blimp consists of three systems, Localization, Power Supply Regulation, and Motor Control Systems (see Fig. 6). These systems have one or more capabilities that are detailed in the following subsections. They assist the operation of different modules essential for autonomous navigation. On the blimp platform, a single downward-looking camera and an IMU sensor are used to capture and stream live-camera feed and odometry data, respectively, through the network to the laptop for further localization processes. Three motors are used to navigate through the construction site. The designed platform also incorporates a power supply regulator. This regulator regulates the voltage to ensure that the components are powered from a constant supply voltage.

3.2.1. Localization system

The blimp localization system consists of a camera and an IMU sensor. This system provides the Control Planning Module with the

Fig. 5. UGV's hardware description.

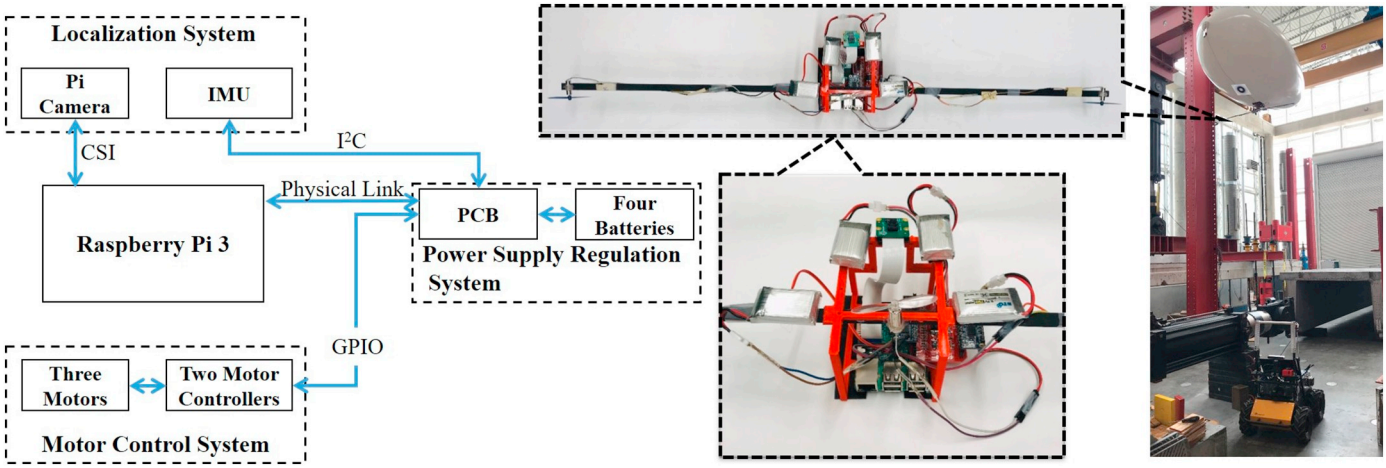

Fig. 6. Blimp system overview (left), payload carrier with all hardware components (right). The channels used for interactions between the different components are labeled in blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

blimp's location and orientation with respect to the starting point. Since the Raspberry Pi 3 is not powerful enough to process this data, it streams the data to the laptop on the UGV through the network. This data are inputs for the SLAM algorithm that estimates the blimp's ego-motion (3D motion of the camera) during its navigation (to be further detailed in Section 4.3.2 Blimp localization using Vins-Mono SLAM algorithm).

A stereo camera is a sensor that captures images together with depth information. However, due to the limited payload capacity of the blimp, carrying a stereo camera with an average weight of 150 g is a challenge. For this reason, a monocular camera that can be lifted by the blimp is used. A Raspberry Pi camera v2 with a weight of 19 g is selected. This rolling shutter camera captures video with the resolution of 410 × 308 with the frame rate of 30 fps, which is streamed to the laptop using a ROS-based video streaming approach [82].

In the localization system, the IMU sensor measures the specific force, angular rate, and the magnetic field surrounding the unit using a combination of accelerometer, gyroscope, and magnetometer. In this study, a Bosch BNO055 IMU with nine degrees of freedom (DOF) is used with the camera to provide accurate information about camera orientation. This sensor offers an absolute orientation (i.e., Euler and Quaternion angles), angular velocity, and linear acceleration at 100 Hz. It uses a high-speed ARM Cortex-M0 processor with minimal power consumption, which is suitable for interaction with Raspberry Pi.

3.2.2. Motor control system

The motor control system forms the basis of the navigation system to control the three-axis motion of the blimp. This system consists of three 3.7-volt DC Hubsan-X4 motors and propellers, one for vertical motion and two for lateral motion. The vertical motion system provides upward and downward movement capability, while the lateral motion system facilitated forward, backward, and left/right (yaw) turn motions. The reason behind having the separated side motors is to generate the maximum torque. Increasing the distance between side motors can generate more rotational power. The IMU provides the yaw at a frequency of 10 Hz, and output is provided to the motors at 10 Hz. The pitch and roll movements are not incorporated in this system. Two motor controller boards (TB6612FNG) can control up to two DC motors each at a constant current of 1.2 A (3.2 A peak). A ROS node provides closed-loop directional input to the blimp. This input is converted into control signals from the GPIO pins on the Raspberry Pi to the motor controllers.

3.2.3. Power supply regulation system

The power of the system is given by four single-cell, Lithium-Ion, 750-milliampere batteries that enable a flight time of 30 min if all three motors are continuously operated all the time. However, the blimp can stay floated without the uplift force from a motor, and the actual flight time can increase significantly. For proper functionality of the motor controllers, sensors, and other hardware components, the system needs to be supplied with a constant voltage. However, the voltage varies over time as the batteries get drained. To address this issue, a linear voltage regulator that receives supply from the batteries is incorporated in the power supply regulation system. The output of the linear voltage regulator supplies a fixed voltage to the components within the system. Two batteries power the Raspberry Pi and sensors. They are connected in series and then regulates to 5 V using an LM1085 voltage regulator. The LM1085-ADJ has a maximum current rating of 3 A. The other two batteries are used to power the motor control system (see Fig. 7).

With the first version of the blimp platform, the system used jumper cables to connect every peripheral to the Raspberry Pi 3. The main problem with this system is that the wiring becomes tedious and difficult to manage. To simplify the system, a custom-built Printed Circuit Board (PCB) is designed using Autodesk EAGLE software [83] to interface various sensors, actuators, and power supply regulation system. It reduces both the complexity (by allowing the sensor systems to integrate into the GPIO pins of the Raspberry Pi simply without large jumper wires) and weight of the system. The PCB integrated the voltage regulator to supply power to the rest of the subsystems, as shown in Fig. 8. Other incorporated components connected to the PCB are two motor drivers, batteries, and an IMU sensor (see Fig. 6).

4. System architecture

This section describes the proposed integrated UAV-UGV system for autonomous data collection on construction sites. This multi-robot system uses ROS to integrate a variety of open-source packages. This architecture enables passing data between different modules, called nodes, within multiple computers through the publisher (i.e., a node that continually broadcasts a message) and the subscriber (i.e., a node

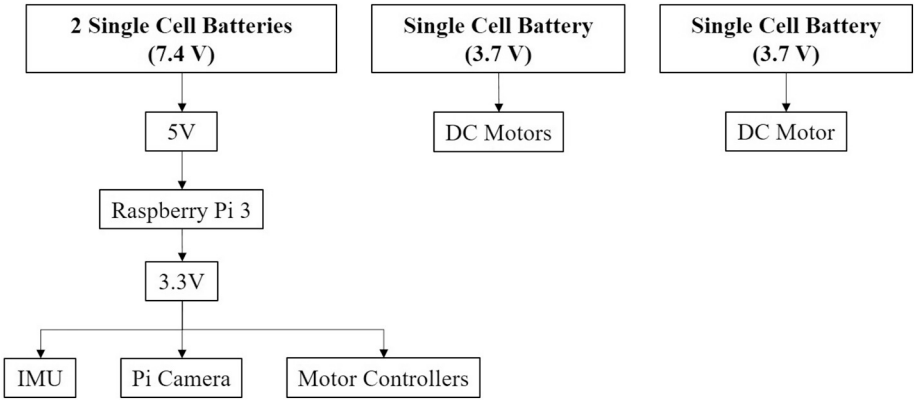

Fig. 7. Blimp power distribution diagram.

that receives a message from a publisher). Each of these modules has many capabilities that are summarized in this section. Fig. 9 illustrates the integration among multiple modules (shown in red boxes).

The system has two separate pipelines for occupancy map generation necessary for autonomous navigation. These maps are generated only by UGV using two independent approaches — one that uses LIDAR to create 2D occupancy maps and the other that uses a stereo camera to generate 3D segmented maps. At this time, the system is able to switch between the two approaches. However, integrating these two maps is not part of the scope of this paper.

In the first pipeline, a LIDAR sensor scans the environment to generate a 2D occupancy map. To avoid the UGV structure interference, the LIDAR's scan range is limited to 180° instead of the full range (i.e., 240 °). The 2D occupancy map, along with the odometry received from wheel encoders, are sent to the Control Planning Module for the path planning process.

In the second pipeline, images and their corresponding depth data from a ZED stereo camera are fed to the Context Awareness Module. A binary scene segmentation scheme (i.e., navigable space and non-navigable space) process the images to generate segmented images. The segmented images, together with their corresponding depth information, are input to the UGV SLAM Module. This module generates a 3D occupancy map of the environment. Besides, it estimates the UGV's position in the generated map at each given time. The Control Planning Module receives this information and relays a free of obstacle path to the Raspberry Pi connected to the ground vehicle for autonomous navigation.

The LIDAR model used in the system is a HOKUYO URG 04LX-UG01. This LIDAR has been designed for indoor use only, and it can scan the environment just in one dimension. The scan area is 180° semicircle with maximum radius 4000 mm. Laser beam diameter is less than 20 mm at 2000 mm [84]. So, the sensor can only map the obstacles which are present at the same height that it is mounted. Although the 2D map generated by the LIDAR sensor is more accurate, the map generated in the second pipeline has the following advantages: (1) It can be used for outdoor environments, (2) In contrary to the map generated by the LIDAR (HOKUYO URG 04LX-UG01), the second pipeline can avoid all the obstacles. Because it maps the obstacles even if they are not present at the same height that the camera is mounted. (3) The generated map using RTAB-MAP is a 3D contextually aware map, where objects of interest (in this study “Ground“ and “Not ground”) are detected, and their real size are estimated.

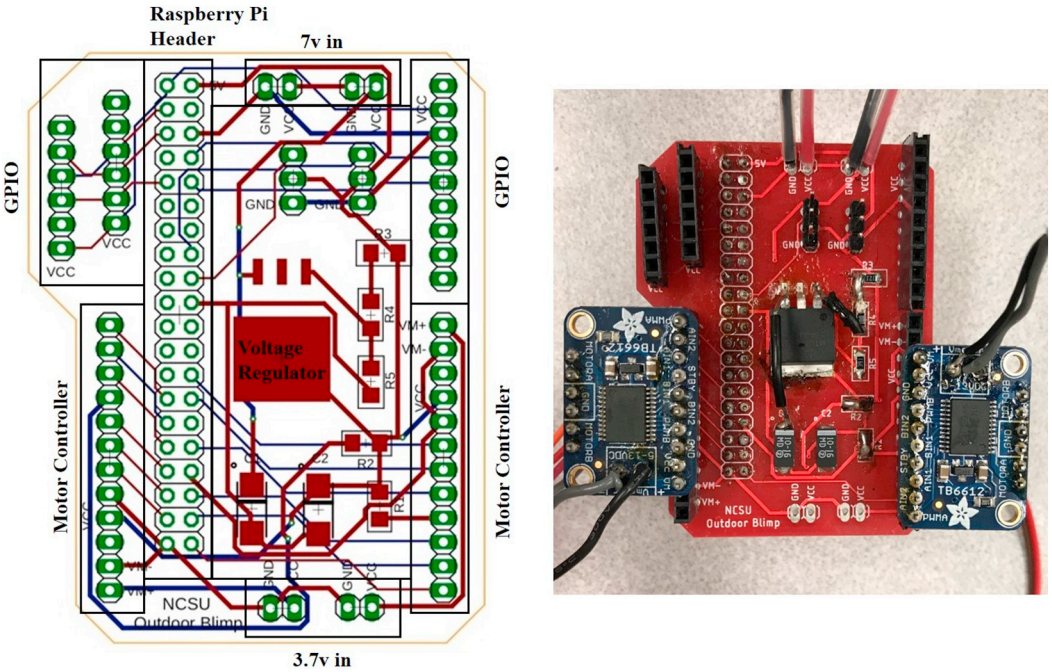

Fig. 8. 3D printed PCB: design (left) and actual board (right).

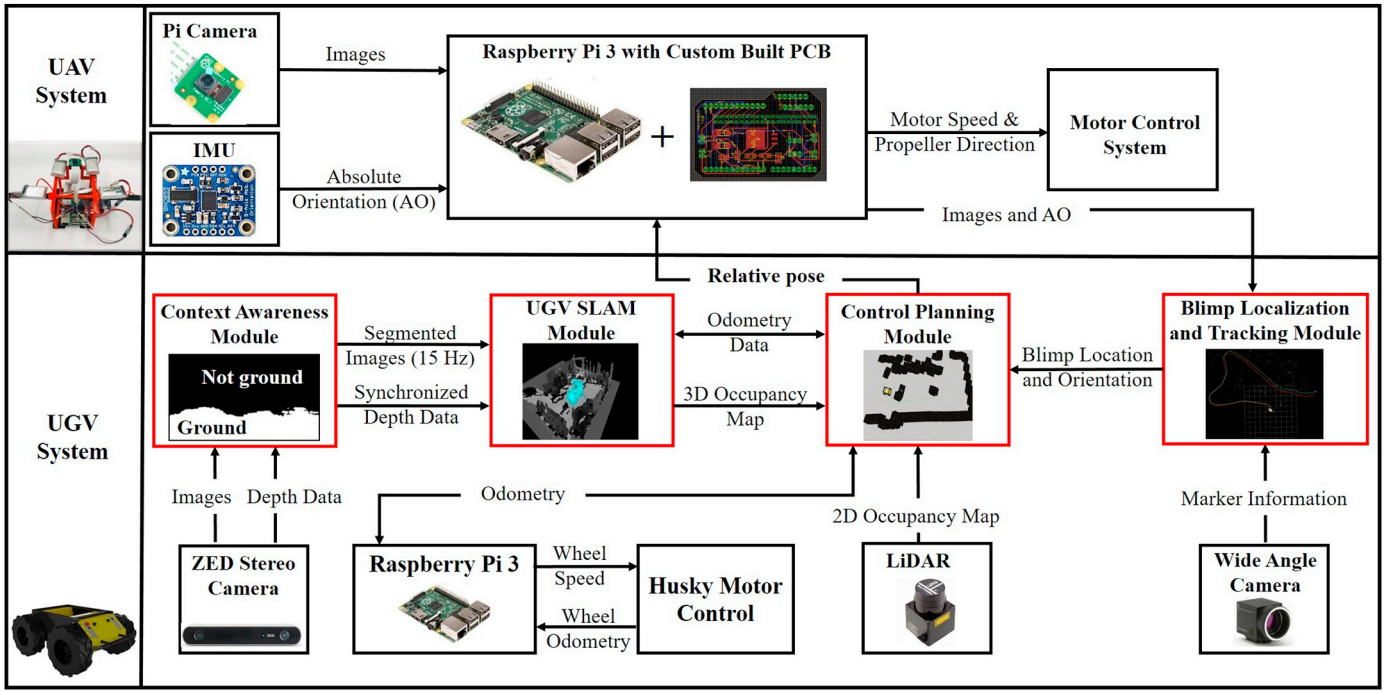

Fig. 9. General pipeline for the proposed integrated system consists of a custom-built ground and aerial vehicles. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

In the initial data collection, the 2D/3D map of the construction site is generated by manually navigating the UGV in the site. During this navigation, the obstacles are detected, and their locations with respect to the UGV are determined. In the further data collections, the destinations are either pre-selected by the construction management team directly through the laptop on the UGV or selected remotely through Secure Shell (SSH), which enables the operator to provide waypoints even during the mission. Then, the Rapidly-exploring Random Tree (RRT) algorithm is implemented to address the point-to-point collision-free navigation problem. In the presence of new obstacles in the UGV's path, the initial map is updated using UGV's sensors. If updated, the RRT algorithm runs immediately to update the navigable path. This capability makes the system robust for navigation in construction scenarios, where navigable paths may dramatically change during a day. Moreover, if some places are not accessible by the UGV anymore (e.g., in the presence of new obstacles/barriers in the environment, which makes the destination inaccessible to the UGV), it sends the blimp to those places for data collection. For this purpose, the relative position of UAV with respect to the UGV within the map is continuously required.

The Blimp Localization and Tracking Module provides the Control Planning Module with the estimated position of the blimp with respect to the UGV. To make this estimation robust, two separate approaches are implemented — marker detection and SLAM comparison. For marker detection, a system consisting of a wide-angle camera is mounted on a tilt unit on the UGV. As the camera detects a marker on the blimp, it sends the relative position between UAV and UGV to the Control Planning Module that enables the blimp to follow the UGV as it is navigating on the site (to be further detailed in Section 4.4.2 UAV autonomous navigation). The tilt unit enables the camera to track the blimp as long as it is in the UGV's field of view (to be further detailed in Section 4.3.1 Blimp localization using a marker). Therefore, the blimp can be localized only if it is in the field of view of the camera.

In the second approach, SLAM comparison, the position of the blimp is estimated by comparing odometry data from the SLAM algorithms that run on the UGV and blimp. The inputs for the SLAM method on the blimp are camera images and absolute orientation (AO) of the camera, captured by the Pi camera and the IMU sensor, respectively. In this approach, there is no need for the UVs to stay close to each other during their navigation. This advantage makes this approach as the primary approach to estimate the relative position. In situations where SLAM tracking is lost for each vehicle, the relative position can be estimated by the first approach. In the following, further details regarding each module are provided.

4.1. Context-Awareness Module

This module uses a ZED stereo camera as a vision sensor. This sensor provides two RGB images for both left and right cameras. It also provides depth information for each pixel in the image. In a depth image, pixels go from black to white. Black indicates that the pixel is close to the camera, and white indicates the pixel is far from the camera. Depth values in the intermediate range are represented by a gradient of gray, where darker gray indicates a depth closer to the camera than lighter colors (see Fig. 10).

The Context-Awareness Module uses this data to provides the segmented image frames alongside with their corresponding depth data to the UGV SLAM Module. Since all the modules are running on the same processing unit, a light segmentation model that focuses on providing a high inference speed and low model size is selected — LNSNet [45]. The LNSNet model takes pixel-wise labeled images as input for training. To the best of our knowledge, no publicly available dataset of semantically annotated images from indoor construction site currently exists. In this work, authors created their own dataset that consists of 3000 images from indoor and outdoor construction site environments. Since the segmented images are finally used for navigation purposes, the training images are labeled in two classes, namely, Ground and Not

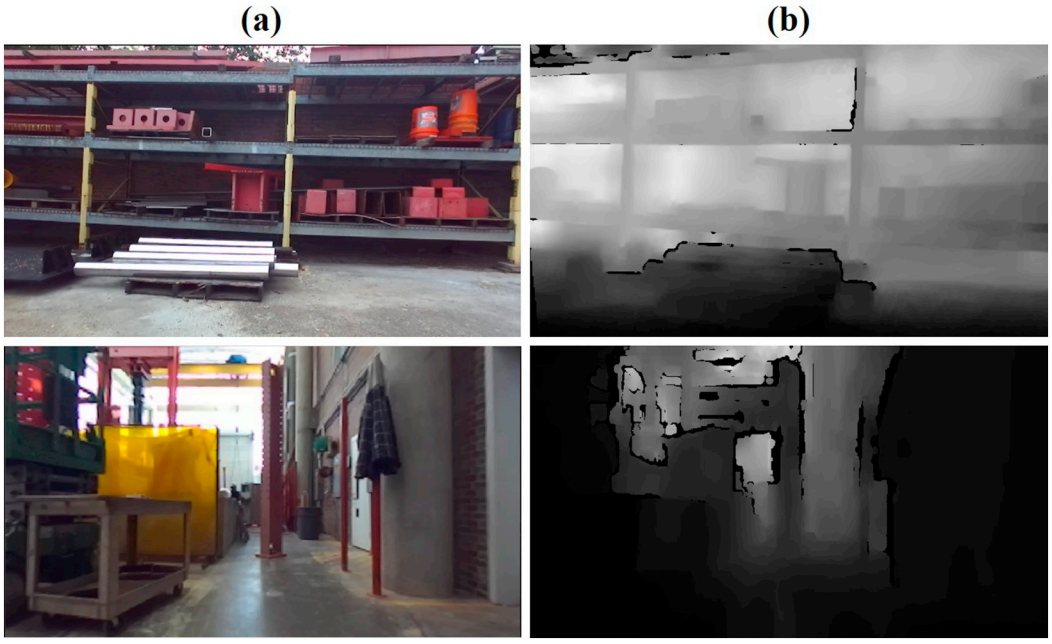

Fig. 10. (a) Raw images, (b) depth images.

Ground. The Ground is indicated by the white pixels and rest by black pixels (see Fig. 11). To extend this approach to accurately segment a wide variety of scenes, the segmentation model needs to be trained with more images from different construction sites.

The model is trained on a server that is equipped with a Xeon E5, 128 GB RAM, and Tesla k40c and Tesla k20c with 12 GB VRAM. The Docker container [85] is used to train the model on the server. Docker is a computer program that performs operating-system-level virtualization, also known as containerization. The Docker virtualization tool is used to overcome different version interdependencies while training on the server. The segmentation model is executed on the Caffe framework [86] with some changes to accommodate the different class labels. The network is trained in 2 steps. The first step involves training the encoder, which takes input of size 640 × 360. In the second step, the decoder is trained on top of the encoder, to upscale the smaller size intermediate map to full image dimensions. The model is trained for 1000 epochs with a batch size of 4. The inference time for the segmentation model running on the laptop for an input image size of 640 × 360 is 21 fps.

The depth data received from the ZED stereo camera is synchronized with segmented images using ROS timestamps. The Context-Awareness Module receives the RGB image frames with corresponding depth data together from the ZED camera using ZED SDK interface [87]. Then, the corresponding RGB frames, depth data, and segmented frames are published as ROS topics with the same timestamp to maintain synchronization. Time synchronization is a necessary step to send corresponding segmented and depth images simultaneously to the UGV SLAM Module.

4.2. UGV SLAM Module

The primary goal of the UGV SLAM Module is to provide the 3D occupancy map using the data received from the Context-Awareness Module while concurrently localizing the UGV within that map. The Control Planning Module uses the occupancy map for autonomous navigation. To achieve this, RTAB-MAP SLAM [42] is implemented to

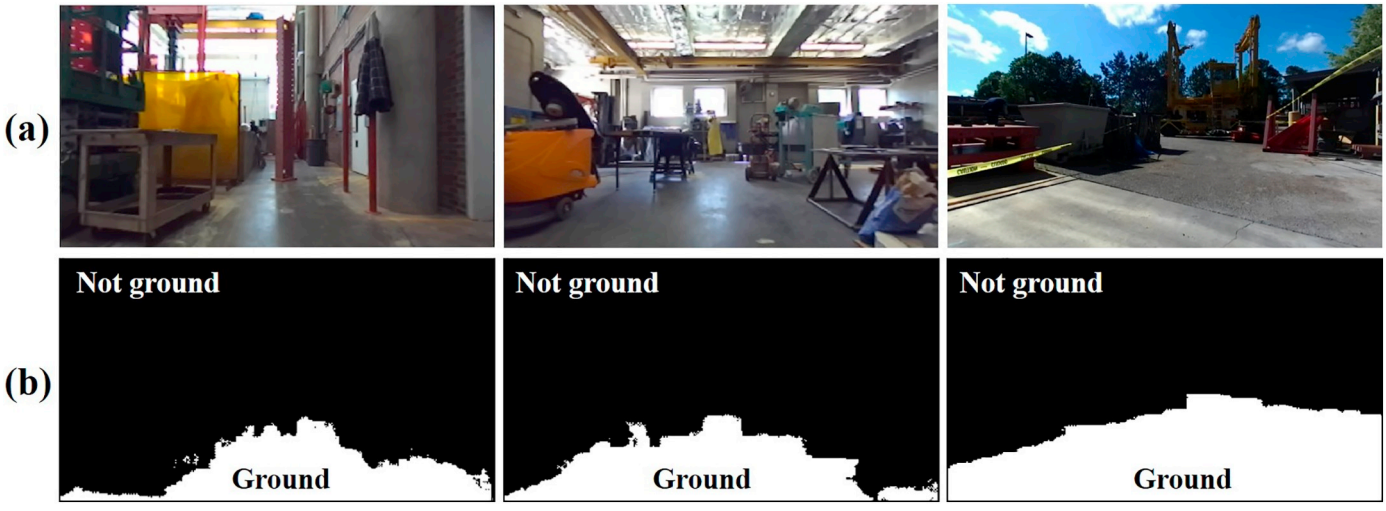

Fig. 11. (a) Raw images, (b) output segmented images.

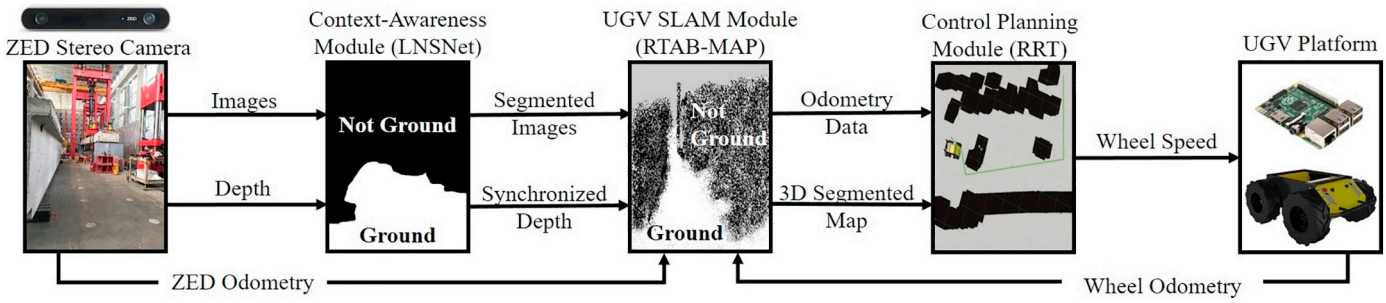

Fig. 12. Autonomous navigation using 3D segmented map provided by the UGV SLAM Module.

generate the 3D segmented map of the environment (3D occupancy map) in integration with the Context-Awareness Module (see Fig. 12). 2D images alongside with depth images from ZED stereo camera are inputs to the Context-Awareness Module. Outputs from Context-Awareness Module (2D segmented images synchronized with depth information) are inputs to the UGV SLAM module. This module processes segmented images with corresponding depth data to create a 3D segmented map of the environment (a 3D map which determines the navigable path with white point cloud and the rest of the scene with black point cloud).

After the initial data collection, the comprehensive map of the environment (known as the global map) is saved to the disk. The global map file consists of the keyframe database, map, and map points, which are serialized to the disk as a binary file using serialization feature of the Boost C++ libraries [88]. This process avoids the need to remap the environment in further data collections by re-localizing in the previously saved map. This mode attempts to find similar visual features by comparing the images the robot currently sees with the images in the database (i.e., new and old image sets, respectively). In the case of navigating in a dynamic environment, this map is updated during further data collections using the UGV sensors. Map updating is done by comparing the odometry data and visual features corresponding to the new and old images. If it is the same, then the new image replaces with the old one and the 3D point clouds created from the receiving images is updated. This capability provides a proper understanding of the environment to the UGV to navigate in a path that is free of obstacles.

The odometry data can be extracted from either the ZED camera or the UGV's built-in wheel odometry. The ZED camera uses visual features and tracks their changes over time to localize the position of the camera with respect to its environment in a technique called 6-DoF pose estimation. The odometry consists of the position of the camera and a vector that represents the camera orientation. These values are not precise enough if the robot faces a feature-less scene (e.g., a wall with a single color) or in the presence of objects too close to the camera, which results in inaccurate localization and mapping. However, it is possible to get a moderately accurate map using this odometry data if the environment contains visual features, and there are not many moving objects passing in front of the camera.

The UGV's wheel odometry can be used as an alternative to the ZED camera odometry. During the experiments, the authors found the wheel odometry to be more accurate. Therefore, during the map generation, this odometry is used instead of the ZED camera odometry, which results in a more accurate map and better localization within the environment. However, in case of eventual wheel slippage, the wheel odometry data is not reliable anymore. To address this issue, the odometry data is reported after processing data from multiple resources, an inertial measurement unit (IMU) and position sensors (incremental encoders). The sensor fusion approach is based on the Extended Kalman Filter (EKF) [89]. EKF efficiently combines the same modality data from multiple sources to provide a more accurate estimation.

4.3. Blimp Localization and Tracking Module

The objective of the blimp is to provide aerial footage that is out of the UGV's view, which involves the blimp to fly to different locations of the construction site. However, collision avoidance against obstacles around the UAV is not part of the scope of this paper. Instead, the UAV uses the relative pose information (i.e., UAV's relative pose with respect to the UGV) to follow the UGV's path as it is navigating in the site. The UGV also sends the UAV to the places which are not accessible to the UGV anymore. It should be notified that these places do not count as unexplored areas. They had been visited by the UGV during the initial data collection before. At the beginning of each data collection, the system re-localizes in the comprehensive map of the site (i.e., global map). This map is generated by the UGV after initial data collection and continuously updated in the further data collections.

During the system navigation for autonomous data collection: 1) the UAV follows the UGV during its navigation, 2) for a place that is not accessible by the UGV, UAV is sent to scan the place, 3) UAV returns on top of the UGV to continue towards the next place of interest. For all these steps, accurate localization is necessary for UAV's self-navigation and relative pose estimation with respect to the UGV. This relative position is continuously sent to the Control Planning Module, which enables the UAV to follow the UGV as it is moving, navigate to the place of interest, and return to the vicinity of the UGV (to be further detailed in Section 4.4.2 UAV autonomous navigation). The blimp localization is performed using two separate approaches, blimp localization using a marker and blimp localization using Vins-Mono SLAM algorithm.

4.3.1. Blimp localization using a marker

An approach similar to Ref. [90] is adapted using a Whycon marker (shown in the yellow box in Fig. 13) that is attached to the blimp to estimate the relative location of the blimp with respect to the ground vehicle. The camera searches for the Whycon marker in each frame as it continuously acquires data of its surroundings. Once the Whycon marker is detected, the relative position is estimated and sent to the Control Planning Module, which enables the blimp to navigate autonomously during the mission. During this process, the blimp needs to be in the camera's field of view, which is achieved by tracking the marker using the camera tilt unit. So, the process of blimp localization using the camera and the marker is divided into two dependent steps, marker localization and tracking unit actuation. Fig. 13 shows the procedure for each step.

Regarding the marker localization step, the inner diameter (

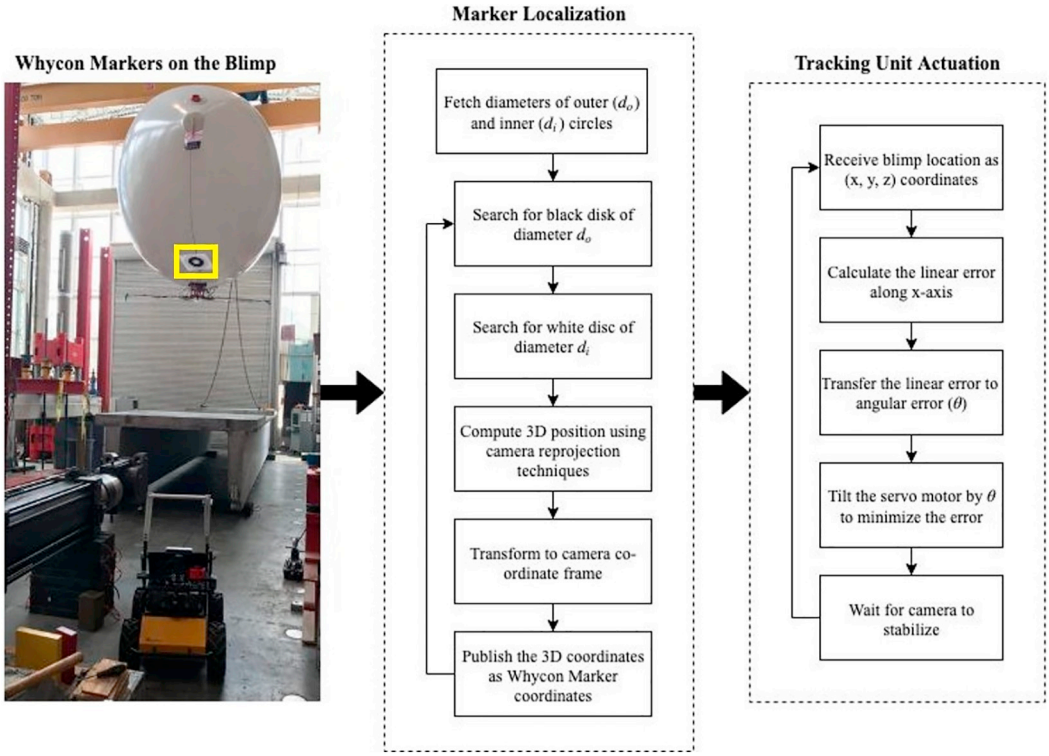

Fig. 13. Blimp localization and tracking procedure.

d_i = 50 mm) and outer diameter (d_o = 122 mm) of the Whycon marker are set as ROS parameters. These parameters act as references for the marker localization algorithm. During this step, first of all, the black disk with diameter d_o is detected. then, the algorithm searches for the white disk with diameter d_i inside the black disk using the flood-fill segmentation technique [90]. The distance of the marker from the camera (i.e., relative position) is then calculated based on the size of the concentric circles depending on the camera re-projection techniques and intrinsic parameters of the camera. This distance is published as the marker coordinates (x,y,z) which is input for the tracking unit actuation step.

The tracking unit is comprised of a servo motor with tilt unit fixed to the camera. This unit receives the (x,y,z) coordinates of the marker and tries to reduce the distance between the center of the camera and the marker in the x-axis(i.e., error along the x-axis). This error is scaled to an angle θ. The servo motor is then tilted by the θ degree which tilts the camera in such a way that the marker is always at the center of the frame and the error is minimum. A video [91] is prepared to illustrate the process of marker localization and tracking unit actuation. Although Whycon marker is used as a marker for the UAV in this paper, other methods of tracking are also available, such as efforts on tracking a 3D moving target [92] or fiducial tags [93]. However, not all of them are compatible with ROS and some require a wrapper.

4.3.2. Blimp localization using Vins-Mono SLAM algorithm

In this approach, Vins-Mono SLAM algorithm [37] is implemented to estimate the location and orientation of the blimp using outputs from Pi camera and IMU mounted on the blimp. Images are streamed at a constant frequency of 30 Hz with 410 × 308 resolution along with the quaternions of gyroscope measurements and linear acceleration along the three Cartesian axes in real-time. In the proposed system, Vins-Mono uses a loosely-coupled sensor fusion, where IMU is treated as an independent module to assist vision-only pose estimation obtained from the visual structure from motion. Fusion is usually done by an Extended Kalman Filter (EKF) [94] where IMU is used for state-propagation, and vision-only pose is used for the update. Further on, tightly-coupled visual-inertial algorithms are either used on the EKF or graph optimization, where camera and IMU measurements are jointly optimized from the raw measurement level.

To have the camera and the IMU sensor on the same coordinate axes, a transformation between the axes of these two sensors is implemented. The data from the IMU sensor and the camera feed are usually not synchronized because of the traffic on the network on the ROS pipeline. The system is set up such that the incoming sensory signals are time synchronized to a tolerance of below 30 ms. This is important as the time skew between the camera and the IMU can have a drastic impact on the camera pose estimation.

Vins-mono is a real-time SLAM algorithm for monocular visual-inertial systems. It uses an optimization-based sliding window formulation for providing high-accuracy visual-inertial odometry. It processes the images to detect key points which are extracted using a Harris corner detector. These points are used to create the history point-cloud, which is saved relative to the camera coordinate system. For each new incoming frame, IMU measurements between two consecutive frames are pre-integrated and previously tracked features are tracked using KLT optical flow algorithm. This algorithm assumes a local neighborhood of pixels for each key point and tracks the change in intensity value by calculating the intensity gradient in x and y directions. VINS Mono gives a very robust localization and a sparse point cloud along with the camera orientation. It can detect the history points and complete a loop closure. Also, the system can eliminate drifts in the path by utilizing a tightly-coupled re-localization module that seamlessly integrates the Visual-Inertial Odometry (VIO). VINS Mono incorporates IMU pre-integration for gyroscope bias calibration, which is then subsequently used for other metric initialization, which results in fast initialization.

4.4. Control Planning Module

This section describes the Control Planning Module approaches for

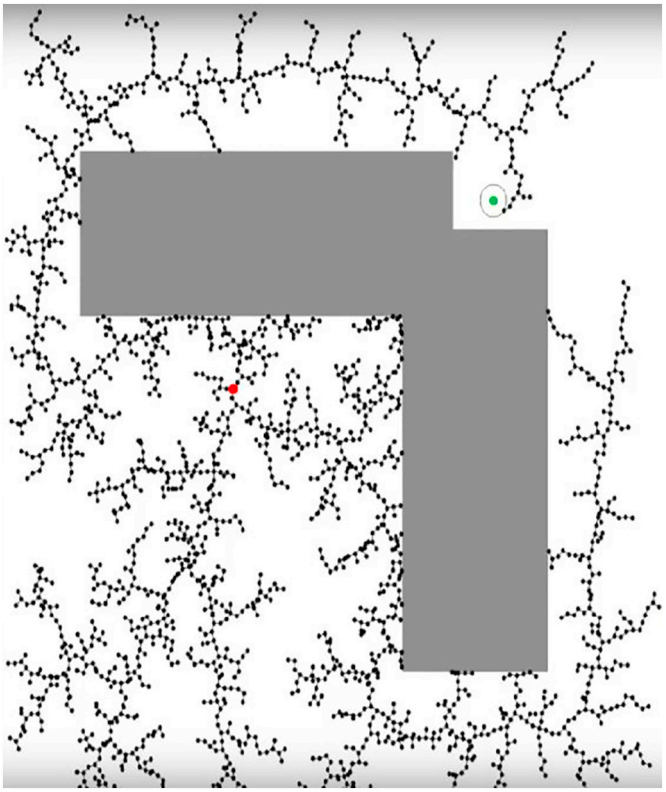

Fig. 14. Visualization of RRT sampling points. The red dot is the UGV's start point and the green dot is the destination. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

both UAV and UGV's autonomous navigation using data provided by other modules.

4.4.1. UGV autonomous navigation

As previously mentioned, the UGV has two separate pipelines for navigation — one that uses LIDAR to generate 2D occupancy maps and the other that uses a stereo camera to generate 3D segmented maps. In both maps, the obstacles are detected, and their location with respect to the UGV is identified. Based on this information, the Rapidly-exploring Random Tree (RRT) algorithm [95] is implemented to address the point-to-point collision-free navigation problem. During the obstacle detection and occupancy map generation, a safety buffer is created around each obstacle. The size of this buffer is almost equal to the UGV's size. The RRT algorithm searches for the possible path free of any obstacles and safety buffers.

RRT is an algorithm designed to efficiently search non-convex, high-dimensional spaces by randomly building a space-filling tree. To achieve this goal, the starting point is set as root. Then a random point is generated on the map, and the node of RRT, which is the closest to the random point is found. A new node at a pre-defined distance from the closest node towards the random point is added to the RRT if the connection path is within a feasible region and the connection is free of obstacles. Otherwise, the new node is added at the edge of the unfeasible region, closer to the RRT node, along the path. The random point generation and linkage to the RRT are performed until the node of RRT hits the target node (see Fig. 14). The resulting path is sent to a path following algorithm to trace the generated path. This algorithm sends the linear and angular velocity commands to the Raspberry Pi, which controls the UGV motors to perform the desired maneuver.

The processing time of this searching algorithm highly depends on the size of randomly generated nodes during the searching process [96,

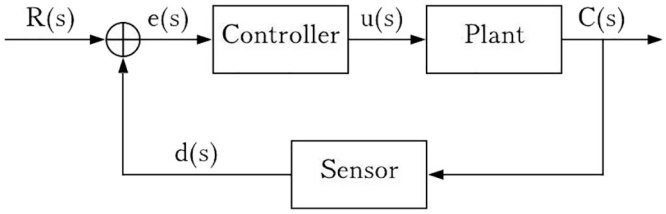

Fig. 15. General control system for the yaw, longitudinal, and lateral axis of motion for the UAV.

97]. In this study, to reduce the processing time, the size of the randomly generated nodes is limited. Tuning this parameter happens through a try and error process until the results (processing time and accuracy) are satisfactory. Considering the environment scattered by obstacles, the size of randomly generated points is limited to 100 to make the process real-time while maintaining the accuracy acceptable.

4.4.2. UAV autonomous navigation

The Control Planning Module enables the UAV to maintain an almost fixed pose relative to the ground vehicle. For this reason, at the beginning of the data collection, the blimp must be calibrated on top of the UGV in a way that both start from the almost same location with the same orientation, but obviously with a difference in Z direction. The relative position result from marker localization approach can be used directly for UAV autonomous navigation. However, for calculating the relative pose between UAV and UGV by comparing the estimated pose from SLAM algorithms running on each vehicle (RTAB-Map for UGV and Vins-Mono for UAV), a transformation matrix between two vehicles is calculated.

With the transformation matrix calculated, the UAV utilizes Algorithm 1 to control the x, y, and z lateral movements as well as the yaw control of the UAV. In this algorithm, the system gets the relative position, orientation (i.e., $[\Delta x, \Delta y, \Delta z, \Delta\theta]$), and predetermined thresholds (i.e., $[d_x, d_y, d_{z\max}, d_{z\min}, d_\theta]$) for these relative values as input. The outputs for each series of input are motor speed and propeller direction. For each input data, based on the current values of relative position, orientation, and corresponding thresholds, the blimp's motors run for a specified duration with a certain speed and direction.

Algorithm 1. General control system for the UAV.

Input:
1 $[\Delta x, \Delta y, \Delta z, \Delta\theta]$: Relative positions in X, Y, and Z axes and orientation.
2 $[d_x, d_y, d_{z_{max}}, d_{z_{min}}, d_\theta]$: predetermined thresholds for relative positions and orientation
3 **foreach** received $[\Delta x_i, \Delta y_i, \Delta z_i, \Delta\theta_i]$ **do**
4 **if** ($|\Delta y_i| > d_y$ *or* $|\Delta\theta_i| > d_\theta$ *or* $|\Delta x| > d_x$ *or* $\Delta z_i > d_{z_{max}}$ *or* $\Delta z_i < d_{z_{min}}$ **then**
5 **while** $\Delta y_i \neq 0$ **do**
6 Rotate the blimp to the UGV location by executing a lateral motor. Move to the UGV (reducing $|\Delta y_i|$) by executing both lateral motors.
7 **end**
8 **while** $\Delta\theta_i \neq 0$ **do**
9 Rotate the blimp to the UGV direction (reducing $\Delta\theta_i$) by executing a lateral motor.
10 **end**
11 **while** $\Delta x \neq 0$ **do**
12 Move the blimp to the UGV (reducing $|\Delta x|$) by executing both lateral motors
13 **end**
14 **while** ($\Delta z_i > d_{z_{max}}$ *or* $\Delta z_i < d_{z_{min}}$) **do**
15 Move the blimp vertically (satisfying $d_{z_{min}} < \Delta z_i < d_{z_{max}}$) by executing the vertical motor
16 **end**
17 **end**
18 **end**

In addition to the algorithm for the lateral movements, a proportional—integral—derivative (PID) control system is used to control

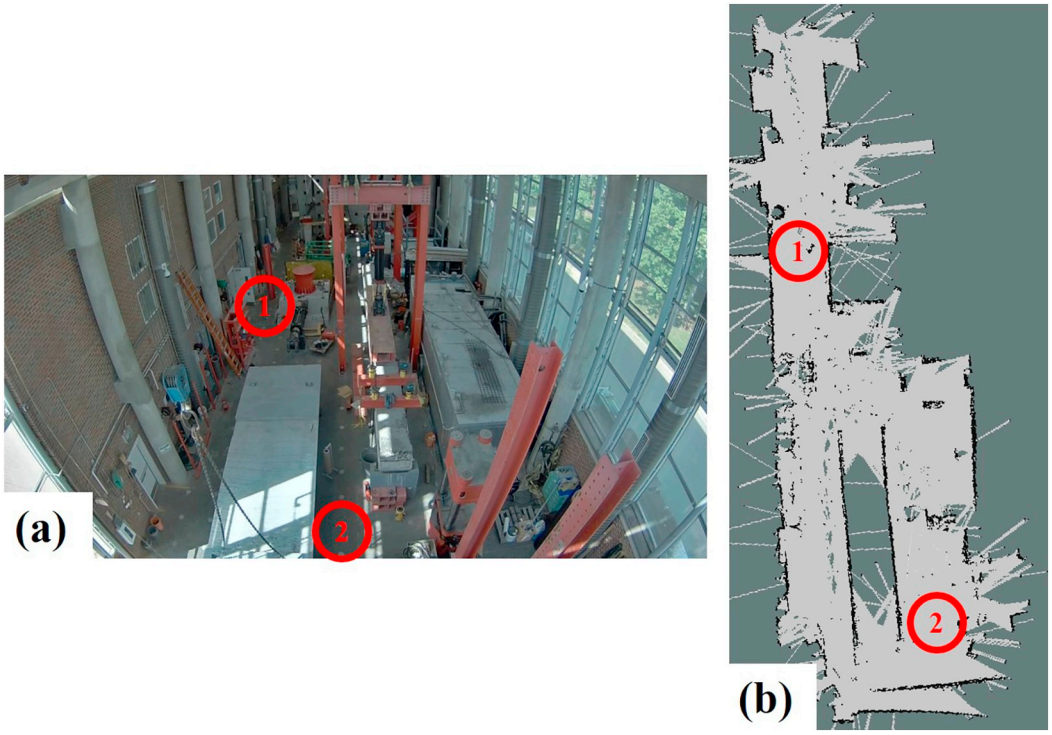

Fig. 16. (a) Test environment, (b) 2D global map. Numbers show the specific locations that the first and second experiments are implemented.

the angular yaw movements in space. A PID control system continuously utilizes the error of the desired set point vs the actual position to calculate the motor power signal. The PID controller manages the motor power to match the UAV yaw angle with the UGV yaw angle. The controller can be expressed as Eq. (1), where u is the motor control variable along a yaw axis; e is the calculated error, and the constant coefficients K_p, K_i, and K_d are tuned based upon the system:

$$u(t) = K_p e(t) + K_i \int_0^t e(t')dt' + K_d \frac{de(t)}{dt} \quad (1)$$

After designing the controller, the PID parameters are tuned for the UAV's angular movements. In order to tune the system, an impulse of a target angle is inputted to the UAV controller, and the output of the magnetometer on the UAV is recorded to analyze the response. The PID parameters are optimized to create a responsive system without overshooting the desired target angle. The general control system is explained by the control loop in Fig. 15, where $R(s)$ is the input signal or the desired set point of the UAV. $e(s)$ is the calculated positional error between the actual UAV position. $u(s)$ is the output of the controller that alters the power of the motors. The Plant is the UAV system. $C(s)$ is the actual position of the UAV. The Sensor is the UAV tracking system on the UGV that calculates the relative position difference between the UAV and the UGV. $d(s)$ is the calculated distance from the sensor. The Controller is a combination of the system described in Algorithm 1 combined with the PID yaw controller.

5. Experimental setup and results

The proposed system has been tested on Constructed Facilities Lab (CFL) at North Carolina State University. This lab environment resembles an indoor construction site as there are stacks of construction materials and ongoing fabrication of structural components (e.g., columns, walls, bridge spans, etc.). Fig. 16 shows the test environments (A and B) and the corresponding 2D global map. Before the experiments, through an initial data collection, a comprehensive map of the construction site (known as the global map) was created by manually navigating the UGV in the site. Then, a destination for data collection is specified within the global map. During the experiment, to deal with a dynamic environment, new obstacles are added to the UGV's path towards the destination. These new obstacles make the rest of the site inaccessible to the UGV. In this situation, the UGV stops and sends the UAV to the desired location to scan the area of interest. The UAV then returns to the UGV, and this integrated team continuous towards the next area of interest.

Videos of the entire pipeline during the tests are prepared to demonstrate the capabilities of this integrated system for indoor data collection and determine if any issues in UGV following is present [98, 99]. Obstacle avoidance for the UAV is not part of the scope of this paper. Instead, the main objective of the UAV is to follow the UGV's path as it is navigating in the site. The UAV is also sent by the UGV to the places which are not accessible to the UGV. Dealing with obstacles around the UAV can be added as a capability to the system in future studies.

During the experiments in the construction site, sometimes the UAV flew close to an air conditioner (AC) vent, and it was blown off course. But, it was able to get back to the desired path to follow the UGV. The focus of validation is the integrated UAV-UGV system consisting of aerial and ground platforms with multiple components. Therefore, this section focuses on the integration and running the system in the real-time, not validations of individual modules (e.g., the accuracy of Vins-Mono, RTAB-Map, and LNSNet). In addition, the UAV autonomous navigation is validated by providing the robots' trajectory alongside with their relative position during their navigation in a path.

The IMU sensors are usually unreliable in the presence of consistent noise in magnetometer measurements (due to heavy metal structures usually present in construction sites). However, it should be notified that, in the Vins-Mono SLAM method, the IMU sensor is used in integration with camera in a loosely-coupled sensor fusion approach, where both the IMU and the camera are treated as independent modules to assist each other. According to the test videos, no significant drift in the trajectory for both the UAV and UGV is observed. In running the system for an extended long period of time, these errors can be

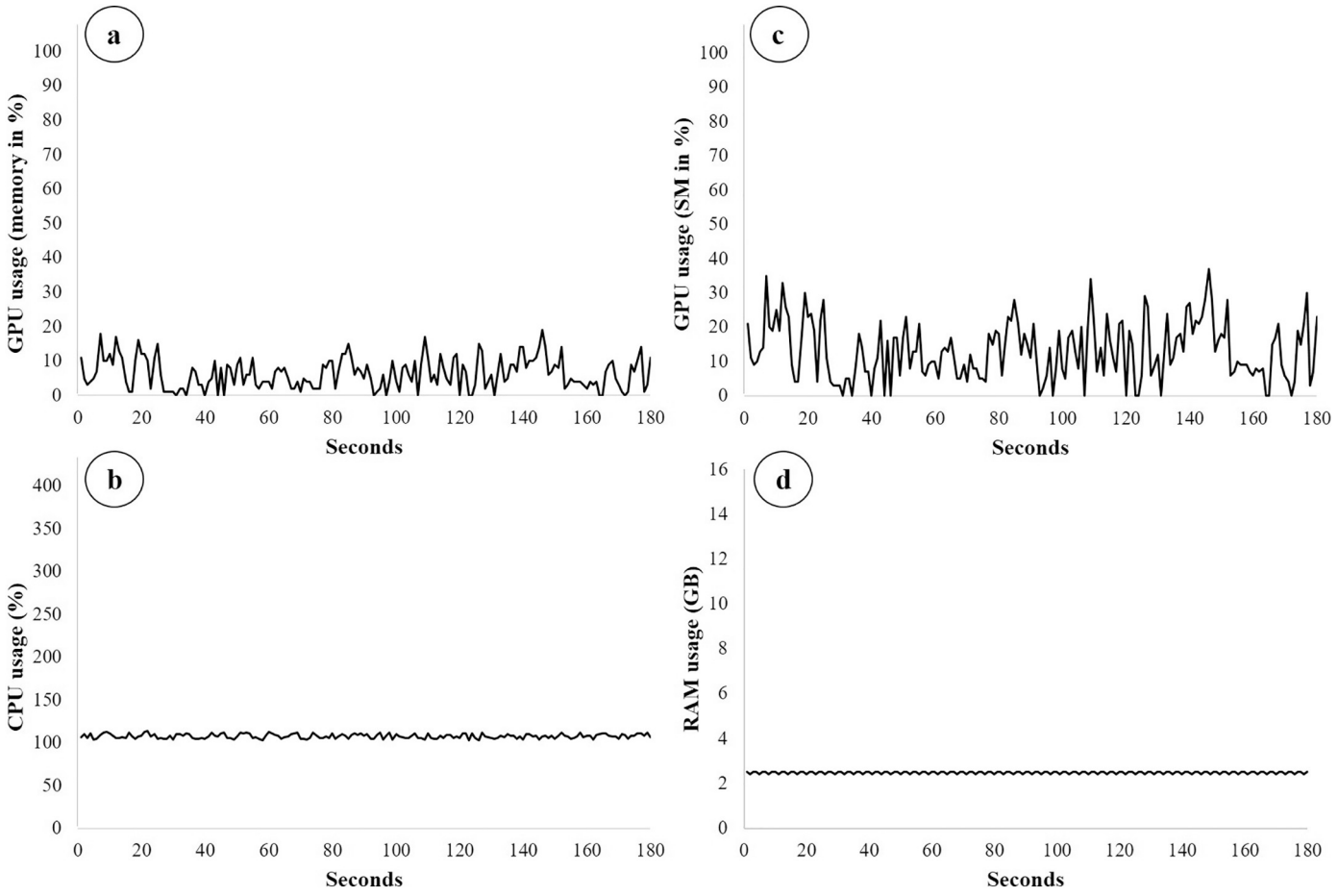

Fig. 17. System usage graphs for semantic segmentation task: (a) LNSNet GPU usage (memory in %), (b) LNSNet CPU usage (%), (c) LNSNet GPU usage (SM in %) (d) LNSNet RAM usage (GB).

Table 2
Semantic segmentation task hardware utilization statistics.

	Min usage	Max usage	Average usage
GPU memory (%)	0	19	7
GPU SM (%)	0	37	14
CPU (%)	103	114	108
RAM (GB)	2.4	2.5	2.45

accumulated and make the results unreliable. However, in such situations, loop closure techniques are activated to optimize the drift errors.

5.1. Hardware utilization

In the proposed system, a single laptop is used to process scene segmentation, localization, mapping, and control planning for both the UGV and UAV in real-time. To evaluate the real-time performance of the system while all the modules with different approaches are running (e.g., both marker detection and SLAM comparison approaches are running to estimate the relative position between the robots), a 3 min experiment is performed. During this experiment, the computational load put in the main processor (UGV's laptop) is recorded using remote logging and reported. Reporting the computational load can be used as a benchmark for future system and pipelines. Running all the modules on a laptop with the specified configuration prevents an unnecessary increase in the number of processing units and processing capabilities. This is a major challenge in developing an autonomous robot because it increases the size and weight, especially with added batteries. It also reduces the operation time with a fixed number of batteries. Moreover, the problem becomes even more challenging if the processes were to be applied to an unmanned aerial vehicle.

RAM utilization is reported in GB unit within a range of 0 –16 as the laptop has 16 GB RAM. The CPU utilization is reported in percentage within a range of 0 –400 as the laptop has a quad-core processor. Therefore, 100% usage corresponds to one core running at its full capacity. For GPU utilization, the percentage of GPU memory (within a range of 0 –100) and Streaming Multiprocessor (SM) are reported. In terms of computational complexity, the Context-Awareness Module, UGV, and UAV SLAM Modules have the heaviest computational load. The results for these modules are reported and discussed in details below.

5.1.1. Context-Awareness Module

The Context-Awareness Module implements LNSNet as the semantic segmentation method. This module uses up to 37% of the GPU memory and the average GPU utilization for inference is 7% as can be seen in Fig. 17a and b, respectively. Table 2 summarizes the findings. The GPU usage is measured using the nvidia-smi [100] utility provided by NVIDIA. The semantic segmentation task is triggers by every input image frame from the ZED stereo camera that takes up some memory. As an image is passed onto the LNSNet network, a binary segmented image is generated and is shown as a spike in the GPU usage in the Fig. 17a and b. After the image is processed, it is released from memory, which explains the regular rise and falls in memory usage (Fig. 17d). The Context-Awareness Module occupies a CPU core running on its full capacity (see Fig. 17c and Table 2).

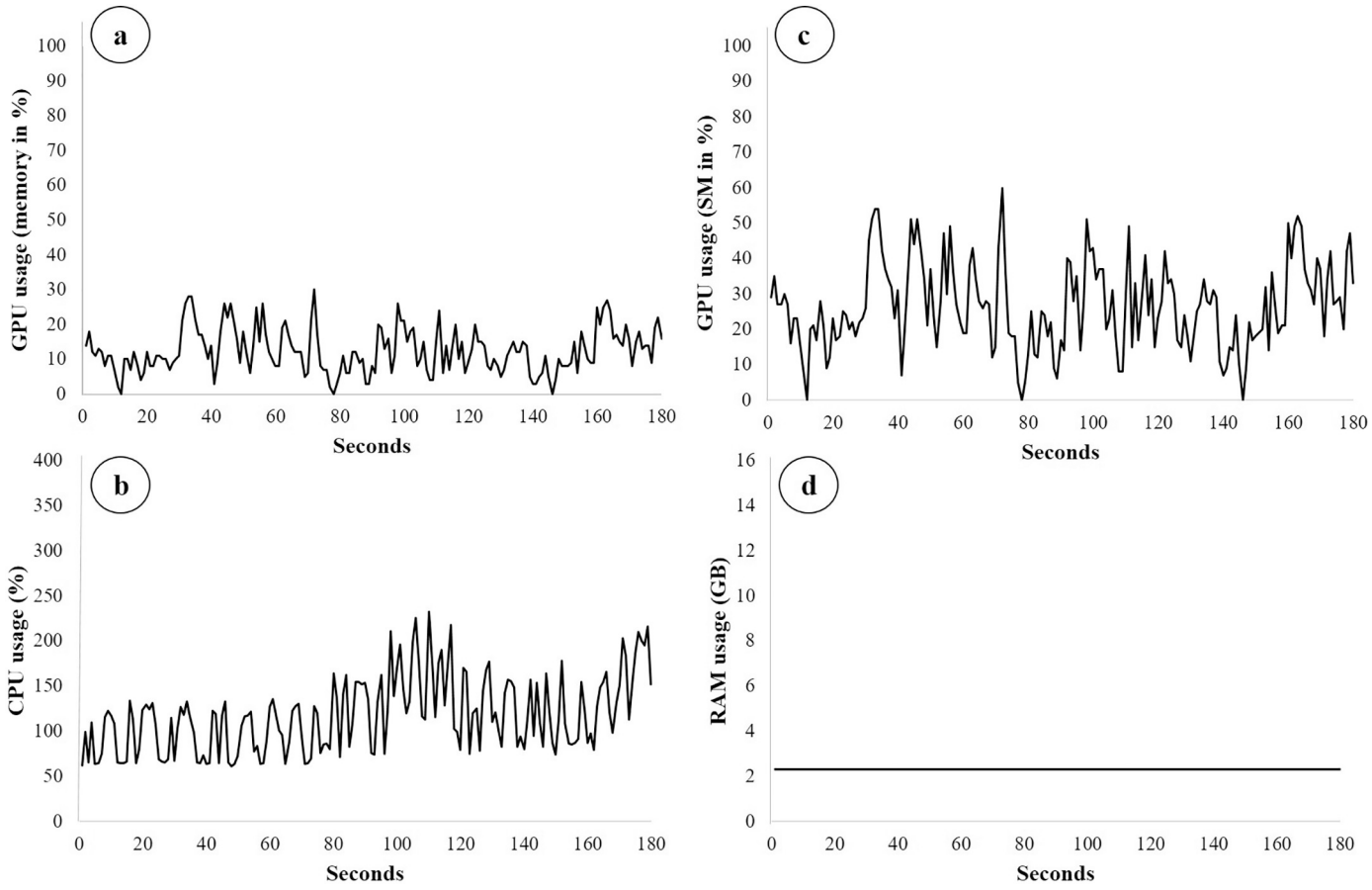

Fig. 18. System usage graphs for ZED stereo ROS wrapper package: (a) GPU usage (memory in %), (b) CPU usage (%), (c) GPU usage (SM in %) (d) RAM usage (GB).

Table 3
ZED Stereo ROS Wrapper hardware utilization statistics.

	Min usage	Max usage	Average usage
GPU memory (%)	0	30	13
GPU SM (%)	0	60	27
CPU (%)	62	233	118
RAM (GB)	2.3	2.3	2.3

The Context-Awareness Module receives input data from the ZED stereo camera through the ZED ROS Wrapper Package [101]. This package enables the camera to be used with ROS. It provides access to the left and right rectified images together with depth information. However, it outputs much unnecessary information for this study, such as point cloud, poses information, and visual odometry. Providing all these outputs as topics results in high CPU and GPU utilization as can be seen in Fig. 18 and Table 3. In the future system, a separate code can be developed to directly get the required topics (i.e., a name that is used to

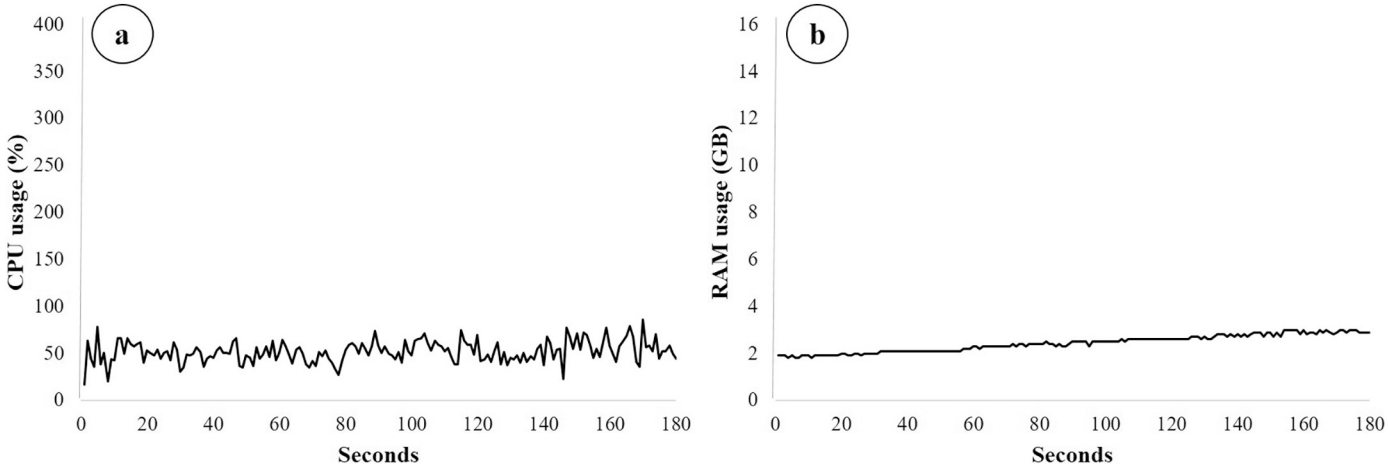

Fig. 19. System usage graphs for RTAB-MAP SLAM algorithm: (a) CPU usage (%), (b) RAM usage (GB).

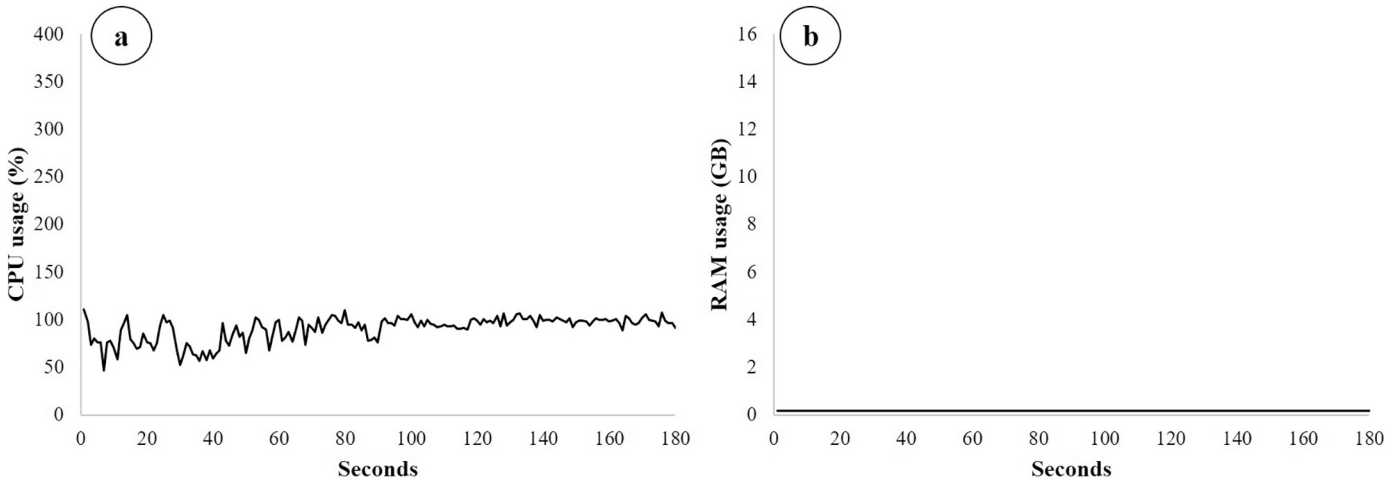

Fig. 20. System usage graphs for Vins-Mono SLAM algorithm: (a) CPU usage (%), (b) RAM usage (GB).

Table 4
RTAB-MAP hardware utilization statistics.

	Min usage	Max usage	Average usage
CPU (%)	17	86	52
RAM (GB)	1.8	3	2.4

Table 5
Vins-Mono hardware utilization statistics.

	Min usage	Max usage	Average usage
CPU (%)	47	111	91
RAM (GB)	0.2	0.2	0.2

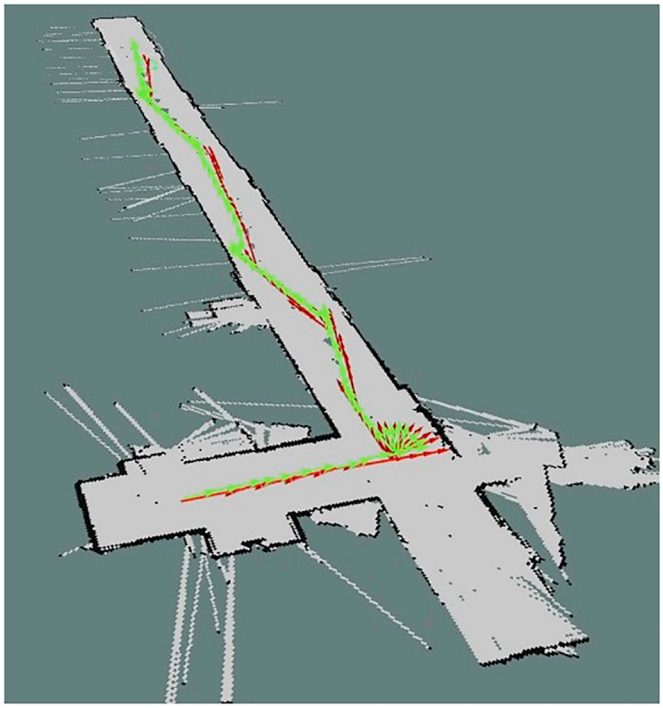

Fig. 21. Poses and orientation shown as arrows: UGV (red) and UAV (green). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

identify the content of the message in ROS) from the ZED software development kit (SDK) which reduces the computational load on CPU, GPU, and RAM.

5.1.2. UGV and UAV SLAM modules

RTAB-MAP and Vins-Mono SLAM run on CPU. As can be seen in Figs. 19a and 20a, CPU usage in Vins-Mono as a monocular SLAM algorithm is higher than usage in RTAB-MAP, which is a stereo SLAM algorithm. Both SLAM algorithms do not use GPU. As can be seen in Fig. 19b, the RAM usage increases as the duration of RTAB-MAP increases due to the increasing size of the point clouds generated in the mapping process. However, the RAM utilization for Vins-Mono is not significant compared to the RTAB-MAP (see Tables 4 and 5). The authors restricted the SLAM algorithm on the UAV to provide only the localization information, and accumulated mapping is not part of the requirements for this system.

5.2. UAV autonomous navigation's validation

To evaluate the performance of the system and determine how well the blimp succeeds in following the UGV, the system is implemented in a hallway that is free of any air disturbance (e.g., AC systems and fans). To make a challenging path for the UAV to follow, the UGV is controlled manually, and multiple rotations along the path are performed. The pose of both the UGV and UAV are visualized in Rviz [102]. Fig. 21 shows a trial run in the hallway for 115 s. At a speed of 0.2 m/s, the UGV and UAV covered almost 21 m. For a longer distance, the speed of the UGV can be increased; however, if the speed is higher than 1 m/s, the UAV will not be able to follow the UGV properly because it receives the movement commands in the frequency of 0.5 Hz. As shown in Fig. 21, as the UGV navigates the hallway, the UAV very closely follows it demonstrating that in a free of air disturbance environment, the UAV can follow the UGV without drifting off the course.

In addition, during the system navigation in the hallway, the relative pose estimated by the marker detection approach (blue triangles) is compared to the relative pose calculated by SLAM comparison approach (orange squares) along *X*-axis, *Y*-axis, and *Z*-axis(Figs. 22, 23, and 24, respectively). As shown in Fig. 25, the relative position is defined as a vector from the UGV sensors (either the ZED camera or the wide-angle camera depends on the approach) to the UAV (either the Pi camera or the marker's center) which satisfies the right-hand rule. In the majority of the times, the UAV moves back of the UGV. So, the relative positions in *X*-axis has mostly negative values (see Figs. 22 and 23). The longest motion is along the *X*-axis(along the direction of travel). Since the hallway is free of any air disturbance, there is no

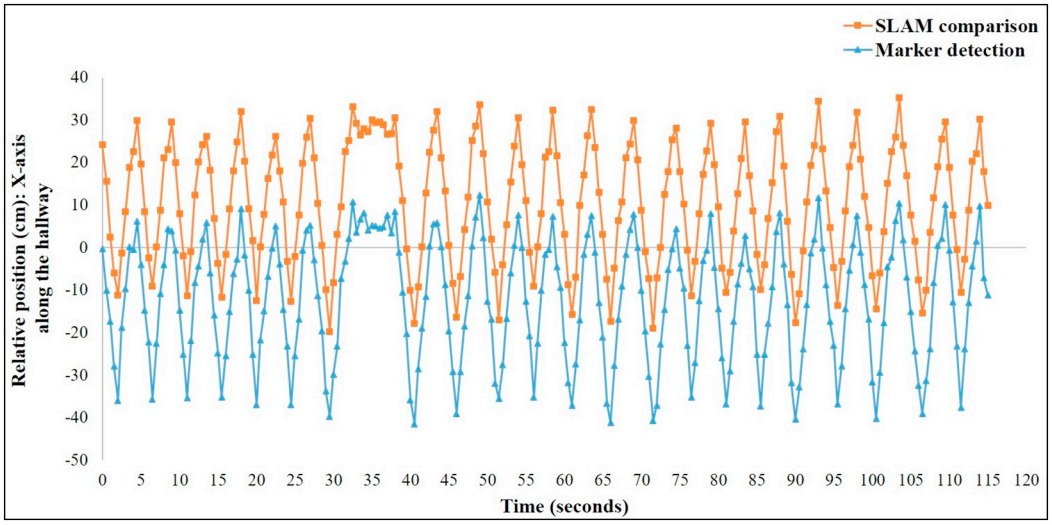

Fig. 22. Relative pose in X-axis estimated by marker detection (blue triangles) and SLAM comparison (orange squares). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

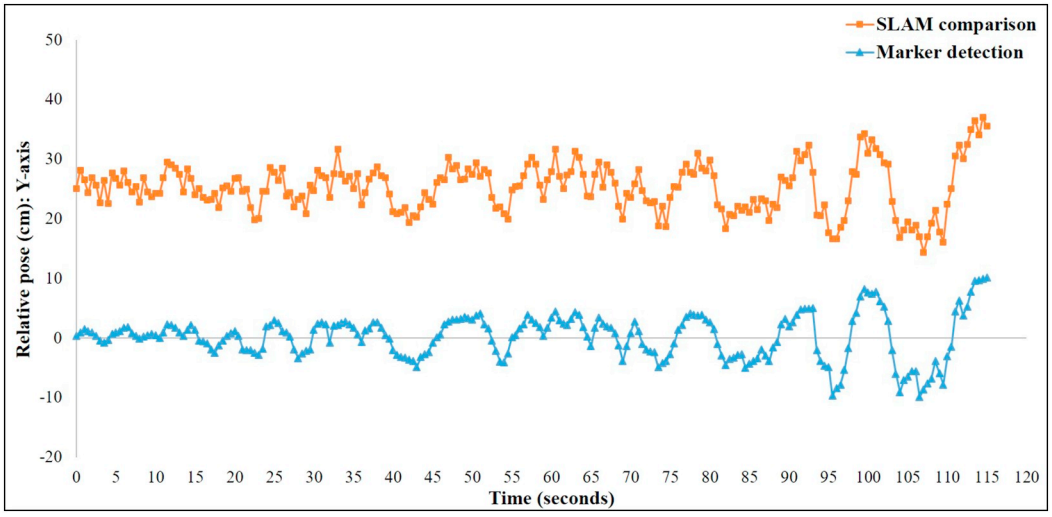

Fig. 23. Relative pose in Y-axis estimated by marker detection (blue triangles) and SLAM comparison (orange squares). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

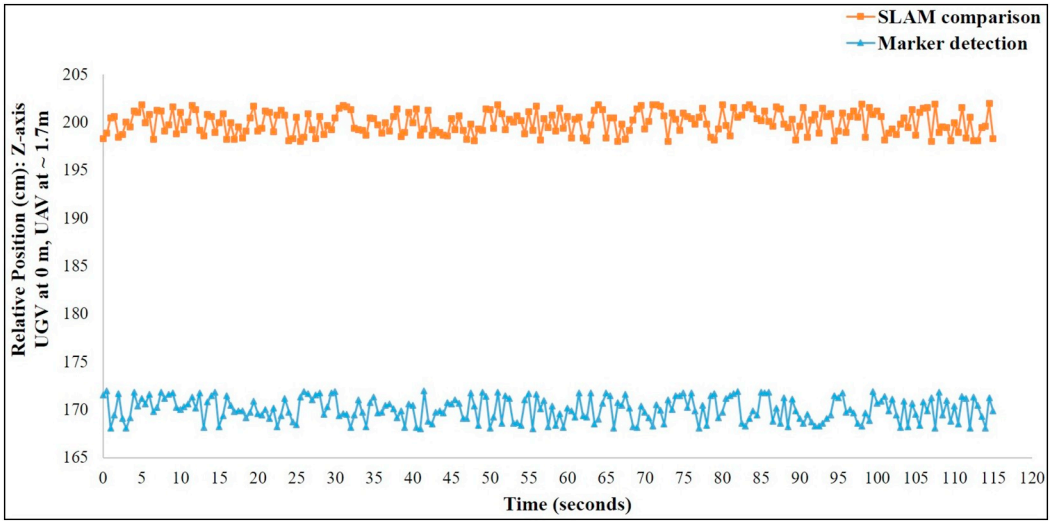

Fig. 24. Relative pose in *Z*-axis estimated by marker detection (blue triangles) and SLAM comparison (orange squares). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 25. Relative pose estimated by SLAM comparison and marker detection approaches.

significant changes in *Y*- and *Z*-axes during the navigation where the UAV stays close to the UGV and maintains a nominal height with small variations (see Figs. 23 and 24). According to the UAV control system for navigation in *X*-axis, as the UGV is moving, if the relative position exceeds a specific threshold (e.g., 35 cm for this experiment), the UAV's lateral motors run in the forward direction to reach the top of the UGV again. These lateral movements are shown as rises and falls in Fig. 22.

The results indicate that there is an almost constant difference (for each axis) between the values obtain by the marker detection and SLAM comparison approaches. The difference is around 25 cm for *X*-axis and *Y*-axis and 30 cm for *Z*-axis. This difference is due to errors in marker detection and distance between the sensors (see Fig. 25), which remain constant (sensor locations are fixed and error in marker detection is around 6 cm [90]). Therefore, to switch from one approach to another, these constant differences (for *x*, *y*, and *z*-axes) are either added to or deducted from the pose values obtained from one of the approaches to compute the values for the other approach.

6. Conclusion and future work

Over the past few years, autonomous UAVs and UGVs have received significant popularity in the construction industry, namely construction site surveying, existing structure inspection, and work-in-progress monitoring. However, there are still numerous open problems for further research such as efficient autonomous navigation in cluttered GPS-denied environments, where some places are inaccessible by the UGVs. To address this issue and increase the degree of automation through vision-based data acquisition, this paper presents a collaborative exploration approach using UAV and UGV that is capable of autonomous navigation, mapping, and data acquisition. The results show that the UGV can autonomously navigate towards the places of interest that are pre-selected by the construction team while being followed by the UAV using vision-based techniques. If an area of interest is not accessible by the UGV due to environmental constraints, the UGV sends the UAV to scan the place. The UAV, then, returns to the top of the UGV and they continue towards the next area of interest. The system is evaluated by performing three experiments (one in a hallway and two in an indoor cluttered construction-like environment). The results are promising, demonstrating the effectiveness and robustness of the proposed system for autonomous navigation and visual data collection for automated construction performance monitoring and condition assessment applications.

Some of the possible extensions and improvements to this study are documented as follows. The hardware utilization can further be improved by preventing unnecessary computations (e.g., useless topics that publish by ZED ROS wrapper node) and reducing the computational load on the CPU by performing heavy calculations (e.g., feature extraction and matching parts in SLAM) on GPU.

The blimp is not stable in the vicinity of active AC systems and fans. Any air disturbance takes the blimp away from the UGV, and it takes some times for the blimp to return to the top of the UGV. However, in the presence of multiple fans and AC vents along the UAV's path, the system cannot be efficient and consumes lots of energy just for stability. The use of fins and tails on the blimps' envelope makes it more stable when facing such conditions. In availability of construction site's BIM model, this model can be integrated with the map provided by the UGV to determine the location of fans and AC vents. Providing this information to the Control Planning Module enables the blimp to skip these locations during its navigation.

Mapping of the environment is performed only by UGV using two independent approaches — one that uses LIDAR to generate 2D occupancy maps and the other that uses a stereo camera to generate 3D segmented maps. However, maps created by the LIDAR and the stereo camera have not been integrated. By integrating these two maps, a more accurate occupancy map is generated, and the system will be able to switch between two approaches. Moreover, future work will perform 3D-mapping of the environment by UAV using its front camera. This map can be combined with the maps provided by the UGV [50], which offers even more accurate map of the environment and improves the UGV's navigation performance for autonomous navigation in unknown environments.

Obstacle avoidance for the blimp is not part of the scope of this paper. Instead, the main objective of the blimp is to follow the UGV's path as it is navigating in the site. Dealing with obstacles around the UAV can be added as a capability to the system by adding more sensors to the blimp such as ultrasonic sensors. Visual-based obstacle avoidance approaches also have the potential to be implemented on the blimp for multi-directional obstacle avoidance. In the future study, multiple approaches will be implemented on the system, and the potential applicability of each method is investigated as the benchmark for the current and future systems.

The proposed system is the authors' first attempt in developing a cooperative UAV-UGV system for autonomous data collection in construction. The focus of this study is on the system's integration and the possibility of real-time applications, not the validation of individual modules (accuracy of RTAB-Map and Vins-Mono algorithms). In the future, work error analysis will perform on each module to determine the effect of individual algorithms and sensor issues (e.g., IMU drifts in the presence of heavy metal structures) on the overall performance of the system. Moreover, techniques such as sensor fusion can be performed to provide results with less uncertainty during experiments running for an extended period of time. For example, odometry data from the camera can be combined with the wheel odometry to provide a more accurate estimation of the UGV pose.

Declaration of competing interest

All authors confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgments

We would like to thank all the students who were part of the ECE 592 class of Spring 2018 for their assistance with this project.

References

[1] T.D. Oesterreich, F. Teuteberg, Understanding the implications of digitisation and automation in the context of Industry 4.0: a triangulation approach and elements of a research agenda for the construction industry, Comput. Ind. 83 (2016) 121–139, https://doi.org/10.1016/j.compind.2016.09.006 http://www.sciencedirect.com/science/article/pii/S0166361516301944.

[2] L.S. Pheng, C.Y. Meng, Managing Productivity in Construction: JIT Operations and Measurements, Routledge, 2018, https://doi.org/10.4324/9780429449116.

[3] T. Bock, The future of construction automation: technological disruption and the upcoming ubiquity of robotics, Autom. Constr. 59 (2015) 113–121, https://doi.org/10.1016/j.autcon.2015.07.022 http://www.sciencedirect.com/science/article/pii/S092658051500165X.

[4] E. Menendez, J.G. Victores, R. Montero, S. Martnez, C. Balaguer, Tunnel structural inspection and assessment using an autonomous robotic system, Autom. Constr. 87 (2018) 117–126, https://doi.org/10.1016/j.autcon.2017.12.001 http://www.sciencedirect.com/science/article/pii/S0926580516303806.

[5] H. Peel, S. Luo, A. Cohn, R. Fuentes, Localisation of a mobile robot for bridge bearing inspection, Autom. Constr. 94 (2018) 244–256, https://doi.org/10.1016/j.autcon.2018.07.003 http://www.sciencedirect.com/science/article/pii/S0926580517303916.

[6] V. Prabakaran, M.R. Elara, T. Pathmakumar, S. Nansai, Floor cleaning robot with reconfigurable mechanism, Autom. Constr. 91 (2018) 155–165, https://doi.org/10.1016/j.autcon.2018.03.015 http://www.sciencedirect.com/science/article/pii/S0926580517306908.

[7] S.-N. Yu, B.-G. Ryu, S.-J. Lim, C.-J. Kim, M.-K. Kang, C.-S. Han, Feasibility verification of brick-laying robot using manipulation trajectory and the laying pattern optimization, Autom. Constr. 18 (5) (2009) 644–655, https://doi.org/10.1016/j.autcon.2008.12.008 http://www.sciencedirect.com/science/article/pii/S0926580509000028.

[8] C. Gosselin, R. Duballet, P. Roux, N. Gaudillire, J. Dirrenberger, P. Morel, Large-scale 3D printing of ultra-high performance concrete a new processing route for architects and builders, Mater. Des. 100 (2016) 102–109, https://doi.org/10.1016/j.matdes.2016.03.097 http://www.sciencedirect.com/science/article/pii/S0264127516303811.

[9] A. Wickowski, “JA-WA” - A wall construction system using unilateral material application with a mobile robot, Autom. Constr. 83 (2017) 19–28, https://doi.org/10.1016/j.autcon.2017.02.005 http://www.sciencedirect.com/science/article/pii/S0926580517301760.

[10] C. Kasperzyk, M.-K. Kim, I. Brilakis, Automated re-prefabrication system for buildings using robotics, Autom. Constr. 83 (2017) 184–195, https://doi.org/10.1016/j.autcon.2017.08.002 http://www.sciencedirect.com/science/article/pii/S0926580517307185.

[11] S. Goessens, C. Mueller, P. Latteur, Feasibility study for drone-based masonry construction of real-scale structures, Autom. Constr. 94 (2018) 458–480, https://doi.org/10.1016/j.autcon.2018.06.015 http://www.sciencedirect.com/science/article/pii/S0926580518301961.

[12] A. Mirjan, F. Augugliaro, R. D’Andrea, F. Gramazio, M. Kohler, Building a Bridge with Flying Robots, Springer International Publishing, Cham, 2016, pp. 34–47, https://doi.org/10.1007/978-3-319-26378-6_3.

[13] K. Asadi, R. Jain, Z. Qin, M. Sun, M. Noghabaei, J. Cole, K. Han, E. Lobaton, Vision-based obstacle removal system for autonomous ground vehicles using a robotic arm, Computing in Civil Engineering, 2019, pp. 328–335, , https://doi.org/10.1061/9780784482438.042.

[14] Y. Ham, M. Kamari, Automated content-based filtering for enhanced vision-based documentation in construction toward exploiting big visual data from drones, Autom. Constr. 105 (2019) 102831, https://doi.org/10.1016/j.autcon.2019.102831 http://www.sciencedirect.com/science/article/pii/S0926580518303698.

[15] K. Asadi, H. Ramshankar, H. Pullagurla, A. Bhandare, S. Shanbhag, P. Mehta, S. Kundu, K. Han, E. Lobaton, T. Wu, Vision-based integrated mobile robotic system for real-time applications in construction, Autom. Constr. 96 (2018) 470–482, https://doi.org/10.1016/j.autcon.2018.10.009 http://www.sciencedirect.com/science/article/pii/S0926580518303625.

[16] Y. Ham, K.K. Han, J.J. Lin, M. Golparvar-Fard, Visual monitoring of civil infrastructure systems via camera-equipped Unmanned Aerial Vehicles (UAVs): a review of related works, Vis. Eng. 4 (1) (2016) 1, https://doi.org/10.1186/s40327-015-0029-z.

[17] J. Park, Y.K. Cho, D. Martinez, A BIM and UWB integrated mobile robot navigation system for indoor position tracking applications, Journal of Construction Engineering and Project Management 6 (2) (2016) 30–39, https://doi.org/10.6106/JCEPM.2016.6.2.030.

[18] K. Asadi, H. Ramshankar, H. Pullagurla, A. Bhandare, S. Shanbhag, P. Mehta, S. Kundu, K. Han, E. Lobaton, T. Wu, Building an Integrated Mobile Robotic System for Real-Time Applications in Construction, Proceedings of the 35th International Symposium on Automation and Robotics in Construction (ISARC), 2018, https://doi.org/10.22260/isarc2018/0063.

[19] K. Asadi, H. Ramshankar, M. Noghabaei, K. Han, Real-time image localization and registration with BIM using perspective alignment for indoor monitoring of construction, J. Comput. Civ. Eng. 33 (5) (2019) 04019031, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000847 https://ascelibrary.org/doi/abs/10.1061/\%28ASCE\%29CP.1943-5487.0000847.

[20] Yuneec, Typhoon H Drones (Last accessed: 01/07/2020), http://us.yuneec.com/typhoon-h-intel-realsense-technology.

[21] Parrot, Discover Parrot's FPV drones (Last accessed: 01/07/2020), https://www.parrot.com/us/drones.

[22] DJI, Drone solution for a new generation of work (Last accessed: 01/07/2020), https://www.dji.com/products/drones-nav.

[23] Y. Fang, Y.K. Cho, S. Zhang, E. Perez, Case study of BIM and cloud-enabled real-time RFID indoor localization for construction management applications, J. Constr. Eng. Manag. 142 (7) (2016) 05016003, https://doi.org/10.1061/(ASCE)CO.1943-7862.0001125.

[24] Y.-H. Lin, Y.-S. Liu, G. Gao, X.-G. Han, C.-Y. Lai, M. Gu, The IFC-based path planning for 3D indoor spaces, Adv. Eng. Inform. 27 (2) (2013) 189–205, https://doi.org/10.1016/j.aei.2012.10.001 http://www.sciencedirect.com/science/article/pii/S1474034612000948.

[25] P. Liu, A.Y. Chen, Y.-N. Huang, J.-Y. Han, J.-S. Lai, S.-C. Kang, T.-H. Wu, M.-C. Wen, M.-H. Tsai, et al., A review of rotorcraft unmanned aerial vehicle (UAV) developments and applications in civil engineering, Smart Struct. Syst. 13 (6) (2014) 1065–1094, https://doi.org/10.12989/sss.2014.13.6.1065.

[26] M. Blsch, S. Weiss, D. Scaramuzza, R. Siegwart, Vision based MAV navigation in unknown and unstructured environments, 2010 IEEE International Conference on Robotics and Automation, 2010, pp. 21–28, , https://doi.org/10.1109/ROBOT.2010.5509920.

[27] M. Zollfer, P. Stotko, A. Grlitz, C. Theobalt, M. Niener, R. Klein, A. Kolb, State of the art on 3D reconstruction with RGB-D cameras, Comput. Graphics Forum 37 (2) (2018) 625–652, https://doi.org/10.1111/cgf.13386 https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13386.

[28] C. Forster, M. Pizzoli, D. Scaramuzza, SVO: fast semi-direct monocular visual odometry, 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 15–22, , https://doi.org/10.1109/ICRA.2014.6906584.

[29] C.D. Herrera, K. Kim, J. Kannala, K. Pulli, J. Heikkil, DT-SLAM: deferred triangulation for robust SLAM, 2014 2nd International Conference on 3D Vision, 1 2014, pp. 609–616, , https://doi.org/10.1109/3DV.2014.49.

[30] J. Engel, T. Schöps, D. Cremers, LSD-SLAM: large-scale direct monocular SLAM, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision - ECCV 2014, Springer International Publishing, Cham, 2014, pp. 834–849, , https://doi.org/10.1007/978-3-319-10605-2_54.

[31] R. Mur-Artal, J.M.M. Montiel, J.D. Tards, ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE Trans. Robot. 31 (5) (2015) 1147–1163, https://doi.org/10.1109/TRO.2015.2463671.

[32] T. Pire, T. Fischer, G. Castro, P. DeCristforis, J. Civera, J. JacoboBerlles, S-PTAM: stereo parallel tracking and mapping, Robot. Auton. Syst. 93 (2017) 27–42, https://doi.org/10.1016/j.robot.2017.03.019 http://www.sciencedirect.com/science/article/pii/S0921889015302955.

[33] R. Mur-Artal, J.D. Tards, ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras, IEEE Trans. Robot. 33 (5) (2017) 1255–1262, https://doi.org/10.1109/TRO.2017.2705103.

[34] D. Galvez-Lpez, J.D. Tardos, Bags of binary words for fast place recognition in image sequences, IEEE Trans. Robot. 28 (5) (2012) 1188–1197, https://doi.org/10.1109/TRO.2012.2197158.

[35] D. Gutierrez-Gomez, W. Mayol-Cuevas, J. Guerrero, Dense RGB-D visual odometry using inverse depth, Robot. Auton. Syst. 75 (2016) 571–583, https://doi.org/10.1016/j.robot.2015.09.026 http://www.sciencedirect.com/science/article/pii/S0921889015002213.

[36] C. Kerl, J. Sturm, D. Cremers, Dense visual SLAM for RGB-D cameras, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 2100–2106, , https://doi.org/10.1109/IROS.2013.6696650.

[37] T. Qin, P. Li, S. Shen, VINS-Mono: a robust and versatile monocular visual-inertial

state estimator, IEEE Trans. Robot. 34 (4) (2018) 1004–1020, https://doi.org/10.1109/TRO.2018.2853729.
[38] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, R. Siegwart, Maplab: an open framework for research in visual-inertial mapping and localization, IEEE Robot. Autom. Lett. 3 (3) (2018) 1418–1425, https://doi.org/10.1109/LRA.2018.2800113.
[39] F. Endres, J. Hess, J. Sturm, D. Cremers, W. Burgard, 3-D mapping with an RGB-D camera, IEEE Trans. Robot. 30 (1) (2014) 177–187, https://doi.org/10.1109/TRO.2013.2279412.
[40] M. Labb, F. Michaud, Appearance-based loop closure detection for online large-scale and long-term operation, IEEE Trans. Robot. 29 (3) (2013) 734–745, https://doi.org/10.1109/TRO.2013.2242375.
[41] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: an efficient probabilistic 3D mapping framework based on octrees, Auton. Robot. 34 (3) (2013) 189–206, https://doi.org/10.1007/s10514-012-9321-0.
[42] M. Labb, F. Michaud, RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation, J. Field Rob. 36 (2) (2019) 416–446, https://doi.org/10.1002/rob.21831 https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21831.
[43] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, (2017) CoRR abs/1704.04861. arxiv: 1704.04861.
[44] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation, (2016) CoRR abs/1606.02147 arxiv:1606.02147.
[45] K. Asadi, P. Chen, K. Han, T. Wu, E. Lobaton, LNSNet: lightweight navigable space segmentation for autonomous robots on construction sites, Data 4 (1) (2019), https://doi.org/10.3390/data4010040 https://www.mdpi.com/2306-5729/4/1/40.
[46] K. Asadi, P. Chen, K. Han, T. Wu, E. Lobaton, Real-Time Scene Segmentation Using a Light Deep Neural Network Architecture for Autonomous Robot Navigation on Construction Sites pp. 320–327, (2019), 10.1061/9780784482438.041, arxiv:https://ascelibrary.org/doi/pdf/10.1061/9780784482438.041, https://ascelibrary.org/doi/abs/10.1061/9780784482438.041.
[47] Y.U. Cao, A.S. Fukunaga, A. Kahng, Cooperative mobile robotics: antecedents and directions, Auton. Robot. 4 (1) (1997) 7–27, https://doi.org/10.1023/A:1008855018923.
[48] B. Arbanas, A. Ivanovic, M. Car, M. Orsag, T. Petrovic, S. Bogdan, Decentralized planning and control for UAV-UGV cooperative teams, Auton. Robot. 42 (8) (2018) 1601–1618, https://doi.org/10.1007/s10514-018-9712-y.
[49] A. Downs, R. Madhavan, T. Hong, Registration of range data from unmanned aerial and ground vehicles, 32nd Applied Imagery Pattern Recognition Workshop, 2003. Proceedings. 2003, pp. 45–50, , https://doi.org/10.1109/AIPR.2003.1284247.
[50] C. Forster, M. Pizzoli, D. Scaramuzza, Air-ground localization and map augmentation using monocular dense reconstruction, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 3971–3978, , https://doi.org/10.1109/IROS.2013.6696924.
[51] S. Hood, K. Benson, P. Hamod, D. Madison, J.M. O'Kane, I. Rekleitis, Bird's eye view: cooperative exploration by UGV and UAV, 2017 International Conference on Unmanned Aircraft Systems (ICUAS), 2017, pp. 247–255, , https://doi.org/10.1109/ICUAS.2017.7991513.
[52] L. Cantelli, M.L. Presti, M. Mangiameli, C. Melita, G. Muscato, Autonomous cooperation between UAV and UGV to improve navigation and environmental monitoring in rough environments, Proceedings 10th International symposium HUDEM, 2013, pp. 109–112, , https://doi.org/10.1061/9780784482438.042 (ISSN 1848-9206).
[53] C. Tucker, J. Tafoyacabrera, C. Montanez, R. Avram, M. Bauer, Y. Jin, An Autonomous Tracking System Integrating UAV and UGVS, http://aseegsw.com/past\%20Proceedings/U-08-Tucker.pdf, (2010)https://doi.org/10.1061/9780784482438.042.
[54] M. Owen, H. Yu, T. McLain, R. Beard, Moving ground target tracking in urban terrain using air/ground vehicles, 2010 IEEE Globecom Workshops, 2010, pp. 1816–1820, , https://doi.org/10.1109/GLOCOMW.2010.5700254.
[55] B. Grocholsky, S. Bayraktar, V. Kumar, C.J. Taylor, G. Pappas, Synergies in feature localization by air-ground robot teams, in: M.H. Ang, O. Khatib (Eds.), Experimental Robotics IX, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 352–361, , https://doi.org/10.1061/9780784482438.042 (ISBN 978-3-540-33014-1).
[56] B. Grocholsky, J. Keller, V. Kumar, G. Pappas, Cooperative air and ground surveillance, IEEE Robot. Autom. Mag. 13 (3) (2006) 16–25, https://doi.org/10.1109/MRA.2006.1678135.
[57] K.A. Ghamry, Y. Dong, M.A. Kamel, Y. Zhang, Real-time autonomous take-off, tracking and landing of UAV on a moving UGV platform, 2016 24th Mediterranean Conference on Control and Automation (MED), 2016, pp. 1236–1241, , https://doi.org/10.1109/MED.2016.7535886.
[58] Cheng Hui, Chen Yousheng, Xiaokun Li, Wong Wing Shing, Autonomous takeoff, tracking and landing of a UAV on a moving UGV using onboard monocular vision, Proceedings of the 32nd Chinese Control Conference, 2013, pp. 5895–5901, , https://doi.org/10.1109/GLOCOMW.2010.5700254 (ISSN 1934-1768).
[59] M.A.G. Oviedo, J.R.P. Valente, D. Zapata, R. Chill, A.B. Cruz, Towards a ground navigation system based in visual feedback provided by a mini UAV, Proceedings of the 2012 International IEEE Intelligent Vehicles Symposium Workshops, 2012, https://doi.org/10.1109/FIT.2012.43 http://oa.upm.es/16245/ (ISBN 978-84-695-3472-4).
[60] R. Al-Jarrah, H. Roth, Ground robot navigation relies on blimp vision information, Int. J. Comput. Commun. Eng. 3 (1) (2014), https://doi.org/10.7763/IJCCE.2014.V3.294 http://www.ijcce.org/papers/294-E034.pdf.
[61] J.H. Kim, Multi-UAV-based stereo vision system without GPS for ground obstacle mapping to assist path planning of UGV, Electron. Lett. 50 (2014) 5895–5901, https://doi.org/10.1049/el.2014.2227 https://digital-library.theiet.org/content/journals/10.1049/el.2014.2227 1431–1432(1).
[62] E.H.C. Harik, F. Guinand, H. Pelvillain, F. Gurin, J. Breth, A decentralized interactive architecture for aerial and ground mobile robots cooperation, 2015 International Conference on Control, Automation and Robotics, 2015, pp. 37–43, , https://doi.org/10.1109/ICCAR.2015.7165998.
[63] P. Fankhauser, M. Bloesch, P. Krsi, R. Diethelm, M. Wermelinger, T. Schneider, M. Dymczyk, M. Hutter, R. Siegwart, Collaborative navigation for flying and walking robots, 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 2859–2866, , https://doi.org/10.1109/IROS.2016.7759443.
[64] H. Kandath, T. Bera, R. Bardhan, S. Sundaram, Autonomous navigation and sensorless obstacle avoidance for UGV with environment information from UAV, 2018 Second IEEE International Conference on Robotic Computing (IRC), 2018, pp. 266–269, , https://doi.org/10.1109/IRC.2018.00056.
[65] Cai Luo, A.P. Espinosa, D. Pranantha, A. De Gloria, Multi-robot search and rescue team, 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, 2011, pp. 296–301, , https://doi.org/10.1109/SSRR.2011.6106746.
[66] N. Giakoumidis, J.U. Bak, J.V. Gmez, A. Llenga, N. Mavridis, Pilot-scale development of a UAV-UGV hybrid with air-based UGV path planning, 2012 10th International Conference on Frontiers of Information Technology, 2012, pp. 204–208, , https://doi.org/10.1109/FIT.2012.43.
[67] E. Mueggler, M. Faessler, F. Fontana, D. Scaramuzza, Aerial-guided navigation of a ground robot among movable obstacles, 2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014), 2014, pp. 1–8, , https://doi.org/10.1109/SSRR.2014.7017662.
[68] G. Christie, A. Shoemaker, K. Kochersberger, P. Tokekar, L. McLean, A. Leonessa, Radiation search operations using scene understanding with autonomous UAV and UGV, J. Field Rob. 34 (8) (2017) 1450–1468, https://doi.org/10.1002/rob.21723 https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21723.
[69] L. Chaimowicz, A. Cowley, D. Gomez-Ibanez, B. Grocholsky, M.A. Hsieh, H. Hsu, J.F. Keller, V. Kumar, R. Swaminathan, C.J. Taylor, Deploying air-ground multi-robot teams in urban environments, in: L.E. Parker, F.E. Schneider, A.C. Schultz (Eds.), Multi-Robot Systems. From Swarms to Intelligent Automata Volume III, Springer Netherlands, Dordrecht, 2005, pp. 223–234, , https://doi.org/10.1007/1-4020-3389-3_18.
[70] J. Peterson, H. Chaudhry, K. Abdelatty, J. Bird, K. Kochersberger, Online aerial terrain mapping for ground robot navigation, Sensors 18 (2) (2018), https://doi.org/10.3390/s18020630 http://www.mdpi.com/1424-8220/18/2/630.
[71] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, E. Takeuchi, S. Tadokoro, Collaborative mapping of an earthquake-damaged building via ground and aerial robots, J. Field Rob. 29 (5) (2012) 832–841, https://doi.org/10.1002/rob.21436 https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21436.
[72] M. Saska, T. Krajnik, L. Pfeucil, Cooperative UAV-UGV autonomous indoor surveillance, International Multi-Conference on Systems, Signals Devices, 2012, pp. 1–6, , https://doi.org/10.1109/SSD.2012.6198051.
[73] H. Qin, Z. Meng, W. Meng, X. Chen, H. Sun, F. Lin, M.H. Ang, Autonomous exploration and mapping system using heterogeneous UAVs and UGVs in GPS-denied environments, IEEE Trans. Veh. Technol. 68 (2) (2019) 1339–1350, https://doi.org/10.1109/TVT.2018.2890416.
[74] S. Batzdorfer, M. Bobbe, M. Becker, H. Harms, U. Bestmann, Multisensor equipped UAV/UGV for automated exploration, Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci. (2017) 33–40, https://doi.org/10.5194/isprs-archives-XLII-2-W6-33-2017.
[75] T. Xu, X. Zhang, Y. Lu, Onboard controlling system design of unmanned airship, Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology, 6 2011, pp. 3028–3031, , https://doi.org/10.1109/EMEIT.2011.6023728.
[76] C. Cheung, S. Singh, M.B. Moseley, B.P. Grocholsky, Integrated long-range UAV/UGV collaborative target tracking, Unmanned Systems Technology XI, 7332 International Society for Optics and Photonics, 2009, , https://doi.org/10.1117/12.820289.
[77] and, A.P. Espinosa, A. De Gloria, R. Sgherri, Air-ground multi-agent robot team coordination, 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 6588–6591, , https://doi.org/10.1109/ICRA.2011.5980582.
[78] M. Garzn, J. Valente, D. Zapata, A. Barrientos, An aerial-ground robotic system for navigation and obstacle mapping in large outdoor areas, Sensors 13 (1) (2013) 1247–1267, https://doi.org/10.3390/s130101247 http://www.mdpi.com/1424-8220/13/1/1247.
[79] ClearPathRobotics, husky-unmanned-ground-vehicle-robot (Last accessed: 01/07/2020), https://www.clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/.
[80] Clearpath, Husky A200 Unmanned Ground Vehicle User Manual, Technical Report, Clearpath, 2011, http://mrsdprojects.ri.cmu.edu/2017teamf/wp-content/uploads/sites/27/2017/10/Husky-A200-UGV-UserManual-0.12.pdf.
[81] StereoLabs, Zed Stereo Camera (Last accessed: 01/07/2020). https://www.stereolabs.com/zed/.
[82] UbiquityRobotics, ROS Node for Camera Module of Raspberry Pi (Last accessed: 01/07/2020), https://github.com/UbiquityRobotics/raspicam_node.
[83] Autodesk, EAGLE: PCB design made easy (Last accessed: 01/07/2020), https://www.autodesk.com/products/eagle/overview.

[84] HOKUYO, Scanning Laser Range Finder URG-04LX-UG01 Specifications, Technical Report, HOKUYO, 2009, http://ftp.k-team.com/KheperaIII/LRF/URG-04LX_UG01_spec.pdf.
[85] D. Merkel, Docker: Lightweight Linux containers for consistent development and deployment, Linux J. 2014 (239) (2014), http://dl.acm.org/citation.cfm?id=2600239.2600241 (ISSN 1075-3583).
[86] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional Architecture for Fast Feature Embedding, Proceedings of the 22Nd ACM International Conference on Multimedia, MM '14, ACM, New York, NY, USA, 2014, pp. 675–678, , https://doi.org/10.1145/2647868.2654889 http://doi.acm.org/10.1145/2647868.2654889.
[87] StereoLabs, Get ZED SDK - Stereolabs (Last accessed: 01/07/2020), https://www.stereolabs.com/developers/release/.
[88] R. Ramey, Serialization for persistence and marshalling, https://www.boost.org/doc/libs/, (2004).
[89] J. Yi, H. Wang, J. Zhang, D. Song, S. Jayasuriya, J. Liu, Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation, IEEE Trans. Robot. 25 (5) (2009) 1087–1097, https://doi.org/10.1109/TRO.2009.2026506.
[90] T. Krajník, M. Nitsche, J. Faigl, P. Vaněk, M. Saska, L. Přeučil, T. Duckett, M. Mejail, A practical multirobot localization system, J. Intell. Robot. Syst. 76 (3) (2014) 539–562, https://doi.org/10.1007/s10846-014-0041-x.
[91] K. Asadi, Marker Localization and Tracking (Last accessed: 07/30/2019), https://youtu.be/R-R3cuaRqow.
[92] A. Chakrabarty, R. Morris, X. Bouyssounouse, R. Hunt, Autonomous indoor object tracking with the Parrot AR.Drone, 2016 International Conference on Unmanned Aircraft Systems (ICUAS), 2016, pp. 25–30, , https://doi.org/10.1109/ICUAS.2016.7502612.
[93] A. Lawrence, A. Turchina, Tag Tracking in ROS (Last accessed: 01/07/2020), https://github.com/ablarry91/ros-tag-tracking.
[94] M. Li, A.I. Mourikis, High-precision, consistent EKF-based visual-inertial odometry, Int. J. Robot. Res. 32 (6) (2013) 690–711, https://doi.org/10.1177/0278364913481251.
[95] S.M. Lavalle, Rapidly-Exploring Random Trees: A New Tool for Path Planning, Tech. rep. 1998 http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.35.1853&rep=rep1&type=pdf.
[96] S. Karaman, E. Frazzoli, Sampling-based algorithms for optimal motion planning, Int. J. Robot. Res. 30 (7) (2011) 846–894, https://doi.org/10.1177/0278364911406761.
[97] S.M. LaValle, Planning Algorithms, Cambridge University Press, 2006, http://planning.cs.uiuc.edu/booka4.pdf.
[98] K. Asadi, Integrated UAV-UGV system for automated data collection in construction (Environment A) (Last accessed: 01/07/2020), https://youtu.be/zbzFNZEwa4Q.
[99] K. Asadi, Integrated UAV-UGV system for automated data collection in construction (Environmnet B) (Last accessed: 01/07/2020), https://youtu.be/yNyddYgQd8c.
[100] NVIDIA, NVIDIA System Management Interface (Last accessed: 01/07/2020), https://developer.nvidia.com/nvidia-system-management-interface.
[101] StereoLabs, Stereolabs ZED Camera - ROS Integration (Last accessed: 01/07/2020), https://github.com/stereolabs/zed-ros-wrappe.
[102] O.S.R. Foundation, Rviz is a 3D visualizer for the Robot Operating System (ROS) framework. (Last accessed: 01/07/2020), http://wiki.ros.org/rviz.