Building an Integrated Mobile Robotic System for Real-Time Applications in Construction

Khashayar Asadi[a], Hariharan Ramshankar[b], Harish Pullagurla[b], Aishwarya Bhandare[b], Suraj Shanbhag[b], Pooja Mehta[b], Spondon Kundu[b], Kevin Han[c], Edgar Lobaton[d], Tianfu Wu[e]

[a,c]Department of Civil, Construction, and Environmental Engineering, North Carolina State University, US
[b,d,e]Department of Electrical and Computer Engineering, North Carolina State University, US
E-mail: kasadib@ncsu.edu

Abstract -

One of the major challenges of a real-time autonomous robotic system for construction monitoring is to simultaneously localize, map, and navigate over the lifetime of the robot, with little or no human intervention. Past research on Simultaneous Localization and Mapping (SLAM) and context-awareness are two active research areas in the computer vision and robotics communities. The studies that integrate both in real-time into a single modular framework for construction monitoring still need further investigation. A monocular vision system and real-time scene understanding are computationally heavy and the major state-of-the-art algorithms are tested on high-end desktops and/or servers with a high CPU- and/or GPU- computing capabilities, which affect their mobility and deployment for real-world applications. To address these challenges and achieve automation, this paper proposes an integrated robotic computer vision system, which generates a real-world spatial map of the obstacles and traversable space present in the environment in near real-time. This is done by integrating contextual Awareness and visual SLAM into a ground robotics agent. This paper presents the hardware utilization and performance of the aforementioned system for three different outdoor environments, which represent the applicability of this pipeline to diverse outdoor scenes in near real-time. The entire system is also self-contained and does not require user input, which demonstrates the potential of this computer vision system for autonomous navigation.

Keywords -

SLAM; Context awareness; Real-time integrated system; Robotic computer vision system; Construction monitoring

1 Introduction

A mobile robot operating in the physical world must be aware of its environment. A large part of this awareness is about estimating spaces (i.e., of mapping) and the robot's location (i.e., localization) [1]. In the absence of external localization aids, the robot must be able to build a map and, at the same time, localize itself in the same partially built imperfect map [2]. The robot must be "contextually aware" of its surroundings, meaning that the robot must be capable of sensing different objects and making situation-specific decisions based on them. This is achieved through object recognition via semantic segmentation, which enables the generation of a spatial map of the obstacles and the traversable space of the environment. This work can also boost autonomous robotic applications to achieve a higher degree of automation in construction monitoring and personalized safety, which have high rate of interest among researchers in this area [3–8].

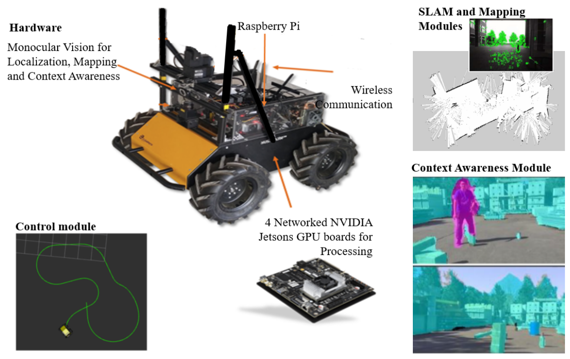
Hardware
Monocular Vision for Localization, Mapping and Context Awareness
Raspberry Pi
Wireless Communication
SLAM and Mapping Modules
Context Awareness Module
Control module
4 Networked NVIDIA Jetsons GPU boards for Processing

Figure 1. Overview of system components- hardware, control, SLAM, context awareness and mapping

The recent advance in powerful and portable processing units have enabled analysis of complex data streams in real-time. These small processing units with high-computing capabilities are well-suited for environmental monitoring using a combination of cameras, microphones, and sensors for temperature, air-quality, and pressure [9–11]. Still, there are a few well-set platforms that combine the state-of-the-art hardware with accessible software and opensources. This paper proposes an integrated mobile robotics agent that is capable of processing localization, mapping, scene understanding, and control and planning. The proposed integrated system uses multiple NVIDIA Jetson TX1 boards [12], each handling a specific task. The low-power consumption and integrated GPU make the Jetson TX1 an ideal candidate for running the aforementioned processes in real-time. As illustrated in Figure 1, the sub-tasks are control, simultaneous localization and tracking (SLAM), image segmentation (denoted as Context Awareness), mapping. For validation, three case studies that captures three different outdoor scenes are performed to evaluate the system's robustness, performance, its integration, and, therefore, its feasibility of the future development of an autonomous ground robot.

This paper is organized as follows: Section 2 presents a literature review on SLAM and scene understanding which are the primary sub-tasks in this approach. Sections 3 describes the system overview and hardware modules of the proposed system. Section 4 describes the approaches taken to perform the above-mentioned pipeline. Section 5 presents the system evaluation and results. Section 6 ends this paper with the conclusions and future works.

2 Background

2.1 Monocular SLAM

In a dynamic environment, human eyes can quickly and accurately provide visual information that our brain uses to map and understand the environment. A robot must also know with a high degree of certainty, where it is located in the environment. Only if this occurs, can it localize itself with respect to the map, which is essential for tasks such as navigation and motion planning. Likewise, using cameras, robots get visual information and they can generate maps of their environment. For building a good map, accurate localization is needed and for accurate localization, a good map is necessary. This chicken and egg problem is what SLAM aims to solve [13].

Initial approaches focused on 2D maps using a laser scanner or LIDAR [14]. The advantage of such approaches was the speed of mapping and the lower computational cost. However, LIDAR-based approaches suffer when there is a lot of heat and reflective surfaces that affect the laser [15]. Also, 2D approaches are unable to capture the scale of the obstacles. With an increase in processing power and development of better algorithms, the use of cameras for SLAM became more feasible.

When it comes to monocular vision-based SLAM, ORB-SLAM [16], Direct Sparse Odometry (DSO) [17] and LSD-SLAM [18] are the widely used algorithms. ORB-SLAM is a feature-based method, while LSD-SLAM is a direct method based on color intensities in the image. Relying on feature extraction, feature matching, and visual odometry, maps are built, but the drawback is that the map is accurate only up to a scale. There are several methods available in the literature which can achieve this task in real-time [16, 18–21]. The approach discussed in [16] is the most appropriate for the current task due to its speed and ability to run in real-time on the TX1. The point cloud and odometry of ORB-SLAM is not directly usable as their units are not in real-world scale. Hence, one of the tasks is to transform the unscaled odometry to real-world units. In this work, the SLAM Module provides the odometry and tracking state to the Context-Awareness and Control Modules.

2.2 Scene Understanding

The literature in object recognition is very rich, and yet growing. The goal of making the robot contextually aware can be achieved through recognition of the objects and taking decisions based on such insights. Either object classification or scene segmentation can be used for scene understanding purposes. Scene segmentation, while being more computationally intensive, provides more precise results, especially near the boundaries of objects.

Even though, convolutional neural networks (CNNs) had been utilized for a long time [22], they seemed to be hard to use for bounding-box object classification up until 2014. This was when an image region proposal scheme was combined with CNN's as classifiers and was able to outperform other object detection frameworks [23]. Later versions of R-CNN object detection were introduced to resolve some of the R-CNN's limitation, such as training pipeline complexity and slow test-time [24, 25]. Fast R-CNN [25] sped up the inference time by a factor of 25. In this method, computation of convolutional layers was shared between region proposals of an image. Faster R-CNN [24] inserted a region proposal network (RPN) after the last convolutional layer. By this change, the method required no external region proposal which improved computational speed up to 250x.

High computational load is one of the main limitations for (CNN)-based frameworks for semantic segmentation. [26] proposed a semantic segmentation framework using fully convolutional networks and utilizing existing classification networks, such as GoogLeNet [27] and AlexNet [28]. This method transfers learning approaches via fine-tuning of pre-trained models. [29] is also based on a very large encoder-decoder model performing pixel-wise labeling which suffers from a tralarge number of computations.

MobileNets [30] and Enet [31] are computationally lighter convolutional networks. ENet is designed to run on embedded boards with a focus on distinguishing roads from the rest of the scene. These capabilities make it suitable for autonomous robot navigation in outdoor (and particularly construction) sites.

3 System Overview

The monocular vision-based approach in the current study is described in this section. Figure 2, shows the

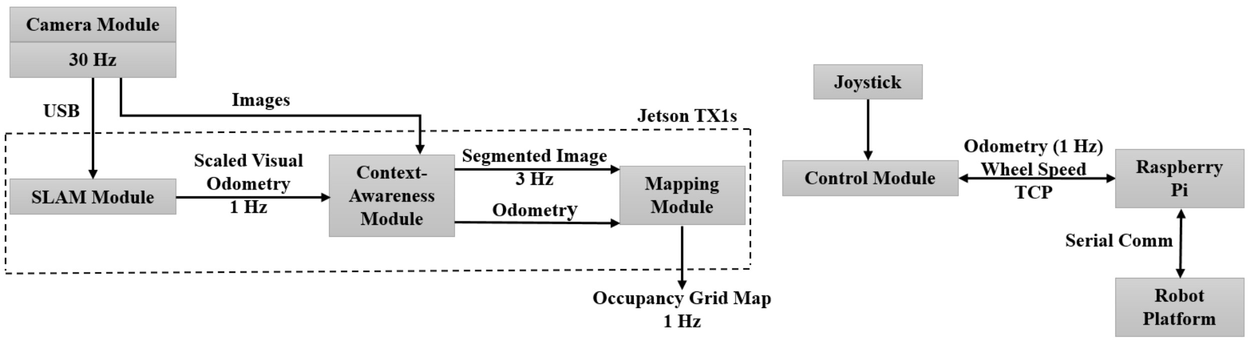

Figure 2. Monocular vision-based general pipeline

integration of aforementioned modules. The SLAM Module sends the scaled odometry to the Context-Awareness Module. The Context-Awareness Module processes the images using a scene segmentation scheme and generates a 1-D array. This array indicates the obstacle boundary which are inputs to the Mapping Module. The Control Module receives the control commands from a joystick and passes them to the Raspberry Pi. Robot Operating System (ROS) [32] is used in this research to simplify the data exchange process between multiple modules.

Figure 3, illustrates the hardware used in the proposed system which are; A Clearpath Husky A200 [33], NVIDIA Jetson TX1 [12], Raspberry Pi, Wi-fi enabled router, and a Microsoft Xbox controller. The controller is connected to the Control Module and is responsible for manually controlling the robot for initial mapping of the scene. This happens by sending this information to the Raspberry Pi through a TCP socket and the Raspberry Pi sends back the wheel encoder information. The Raspberry Pi can also operate a kill switch to stop the motors in case of an emergency. In Figure 3, There are four NVIDIA Jetson TX1 boards. Each module runs on one board to minimize logistics and integration time. A network via router on the robot connects all the Jetson boards.

4 System Description

The main goal of this research is the integration between SLAM Monocular Module, Context-Awareness Module, Mapping Module, and Control Module. This integration leads to the generation of an occupancy grid map of the environment which forms a spatial map of obstacles and traversable space of the scene. In this section, other capabilities of these modules are explained in detail.

4.1 Real-time visual SLAM with scaled odometry

The provided odometry to the Context-Awareness Module cannot be used directly. To solve this issue, the proposed approach in [34] is implemented to get the scaled information between visual odometry and wheel odometry. Both odometry are calculated for the entire path and then a closed-form solution is generated. Finally, The scaling matrix between wheel odometry and visual odometry is found and updates every second. The final odometry output that other modules can use is published as a ROS topic at a frequency of 1 Hz, as shown in Figure 1. Figure 4, shows scaled ORB-SLAM in red compared to unmodified SLAM in blue which shows that the scaled ORB-SLAM improves the trajectory. The unmodified SLAM is not able to detect the simple turn or even moving in the straight line.

4.2 SLAM and Context-Awareness Modules contribution to provide segmented images

The next step is to provide images and synchronize scaled odometry to the Context-Awareness Module for the segmentation of ground plane and obstacles. Contextual Awareness Module provides the intelligence and aids in decision-making while in motion. It strives to improve a general awareness of the environment by enhancing the visual information from the monocular camera.

Creating the segmented images and preparing them to be used by Mapping Module consists of three steps; pixel-wise semantic segmentation, filtering the segmentation vector, and perspective transformation. The segmentation model Enet [31], is used to produce a pixel-wise semantic segmentation map per image. The segmentation vector is a 1-D vector along the horizontal axis that represents the distance to the closest object at each point. Finally, a perspective transformation is implemented to convert from image to the world coordinate system. This information is used by Mapping Module to create the occupancy grid map of the environment.

4.2.1 Pixel wise semantic segmentation

The pixel-wise labeling task assigns a label to every pixel of the image. This typically requires models that are computationally heavy, with a lot of parameters. Due to

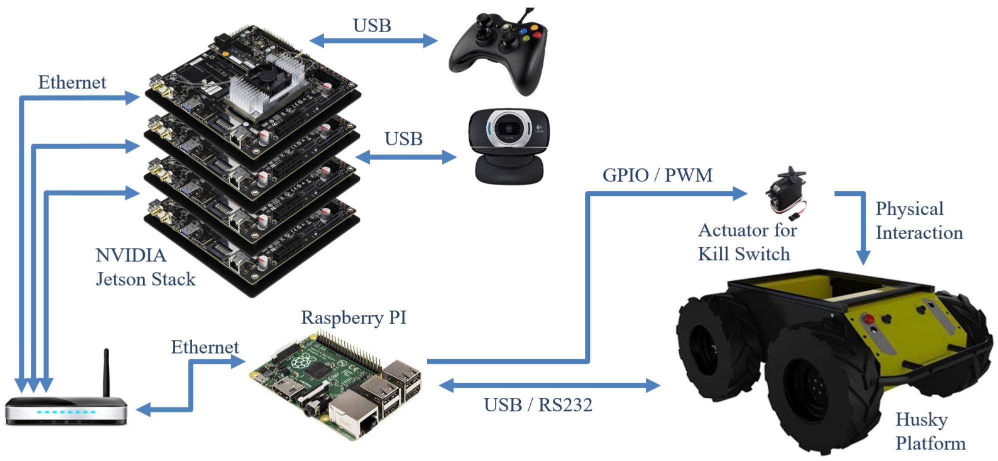

Figure 3. Physical diagram of components in the platform. The channels used for interactions between the different physical modules are labeled in blue.

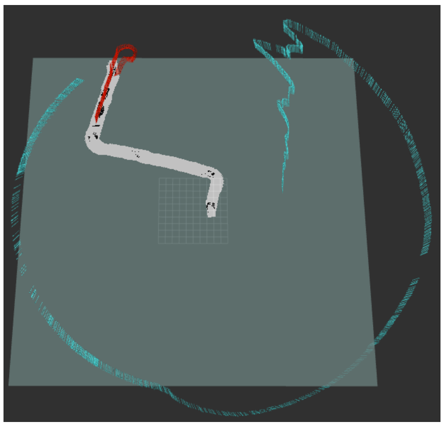

Figure 4. Scaled ORB SLAM odometry (red) vs. unscaled ORB SLAM odometry (blue)

the limited computing capability of the Jetson, a smaller model (ENet) is chosen with a slightly lower accuracy, but fast enough for near real-time performance. ENet method for semantic segmentation [31] is designed to work on embedded boards and in the current research this method is implemented on the Jetson TX1 with image input size of 512×256, at speed of 10 fps (much faster than other models such as Segnet and FCN [26, 29]).

The labeled pixel-wise data are input to ENet during the training phase. In this research, a scripting file is implemented to easily label images with freehand drawing. 1000 images, captured from multiple videos, are sampled and labeled manually. Grey-scale images with pixel information being the class label and size similar to input image size are used for training the network. Five labels are used for labeling as shown in Figure 5: object, road, person, sky, and unlabeled.

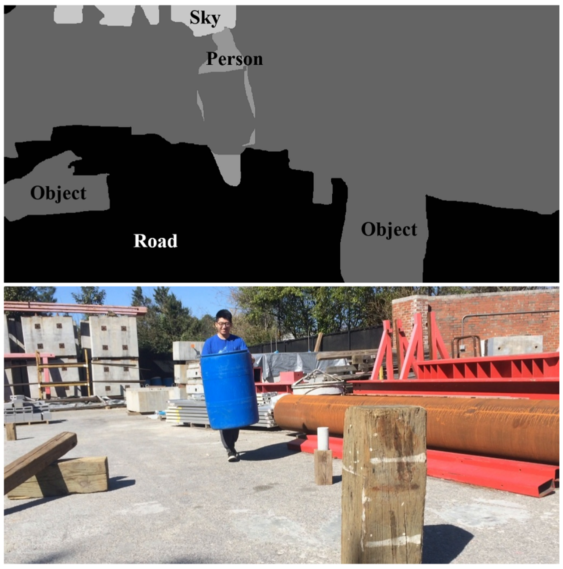

Figure 5. Different labeled classes

The training process includes 2 steps. First, training the encoder part. The input images for encoder training has a size of 512×256 and the size of output labeled map is 64×32. The model is trained for 300 epochs with a batch size of 10. Second, training the decoder part on top of the encoder to convert the intermediate map into the same dimensions as the full image.

4.2.2 Filtering the segmentation vector

The robot can move safe in the places that are known as road in the image. First of all, a 1-D vector is computed which provides the first instance of the obstacle when going through the bottom (the blue line in Figure 6). The filter is tuned to only include objects that are within 2.5 meters of the camera. Also when the vector is vertical it probably means that it is not actually the base of the object so it should not be included in the occupancy grid. To this end, points with the high gradient (more than one) in the x-direction are filtered out. The red segments in Figure 6, indicate filtered segments which are plotted as obstacles.

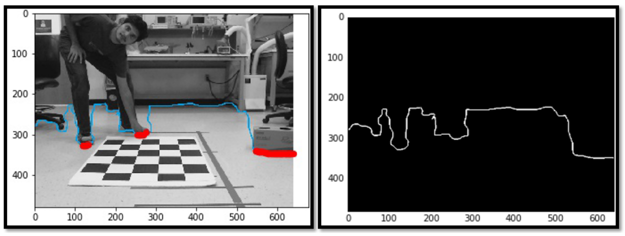

Figure 6. Red dots indicated filtered points to be considered obstacles

4.2.3 Perspective transform

The 2D image coordinates are transferred into physical distance values from the base of the robot using perspective transform. In the proposed system, the location and orientation of the camera are fixed related to the robot. Hence, a fixed perspective transform from specific focal length and camera position can be used for the whole process of mapping the pixel locations to its real-world locations (see Figure 7).

In reality, the perspective transform matrix is little different with the transformation matrix in Figure 7. The reason is, the origin starts from the top of the image, not the camera itself. Also, it is necessary for the pixel mapping to be symmetric on the left and the right sides of the robot, but the camera is not exactly in the center of the rectangle.

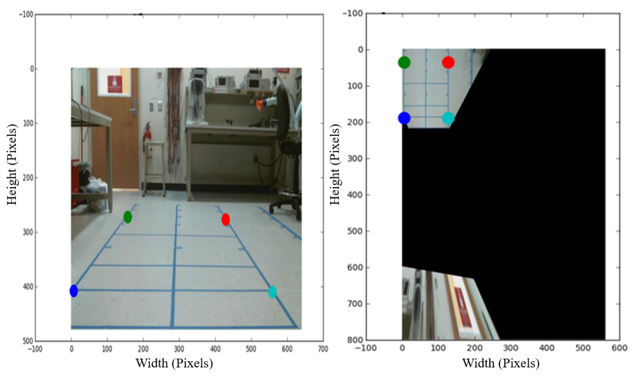

Figure 7. Left: Original image, Right: Perspective transform

4.3 Occupancy grid map creation

The Mapping Module uses the segmentation results to provide an occupancy grid map. The aforementioned obstacle position vector from Context-Awareness Module is processed to find the lower boundary of close obstacles. Next, the perspective transformation matrix helps to find the real world obstacle locations. Finally, the position of obstacles is plotted on a local rectangular map with 1.1m wide and 2.5m long and the global map incrementally updates by using the local map (see Figure 8).

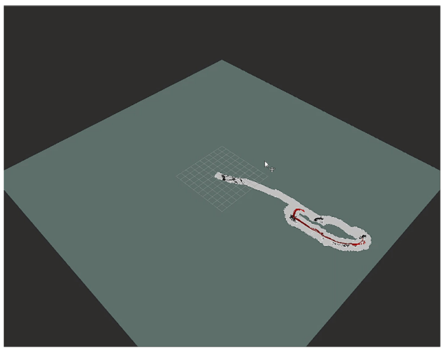

Figure 8. Global map

4.4 Publishing tracking state of ORB-SLAM for SLAM and Control Modules integration

It is necessary for the SLAM Module to know whether or not ORB-SLAM has initialized tracking. The Context-Awareness Module is not able to segment the images if the SLAM Module loses track. Also, the Mapping Module needs the images from SLAM Module to update the global map and the SLAM Module can only send the images in the tracking state. SLAM Module publishes following states: waiting for images, not initialized, tracking, and tracking lost. Providing this information for Control Module enables the robot to retrace its path in case tracking was lost. [35] shows the tracking state in the ORB-SLAM Module.

5 System Evaluation and Results

The proposed system is tested in three different outdoor environments with various object and weather conditions as shown in Table 1. The Figure 9 shows a representative image of the environment and a screenshot of the occupancy grid map during its generation. The RVIZ tool of ROS is used to visualize the occupancy grid map. The red line in the image is the trajectory of the robot and the grey rectangles represent the mapped areas.

The focus of the validation in this research is on the integrated robotic system which brings multiple components together and runs in real-time. The videos of the

Table 1. Environment description

Evironment Type	Object Type	Weather Condition	Video Length (minutes)	Number of processed frames (pipeline rate of 1 Hz)
Parking space 1	Car, curb	Cloudy	12.7	762
Construction site	Wooden planks, trash bin, cement slab	Cloudy	15.15	909
Parking scene 2	Trash bins, utility cart	Sunny	9.5	570

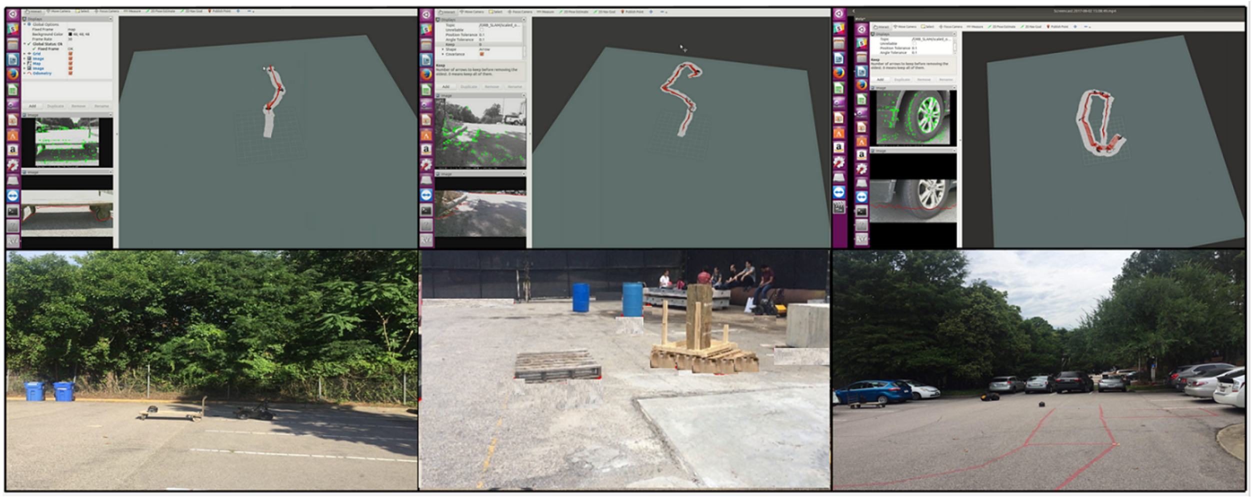
Figure 9. Image of the environment (bottom) and its corresponding occupancy map (top).

whole pipeline demonstrate the capabilities of this integrated system in near real-time [36–38].

The computational load on the Jetson boards is also presented for future systems and pipelines (see Table 2). Since the Jetson board has a quad-core processor, the percentages for the CPU are within a range of zero to 400. The CPU usage of ORB-SLAM is very high, but the scaling process in SLAM Module requires light computing. Figure 10, shows that the Context-Awareness Module runs heavily on the GPU. When an image is passed over to the ENet network, a segmented image is produced and is shown as a spike in the GPU usage in Figure 10. This process publishes the boundary-position vector at the end of processing. The hardware usage for Mapping Module is not significant and it shows the potential of implementing more than one module on one Jetson.

Table 2. CPU usage statistics. (100 corresponds to full usage of one core)

Code/CPU Usage	Average CPU Usage
ORB SLAM	186
odometry scaler	5
segmentation process	101
store_images	33
global Map	54
local Map	18

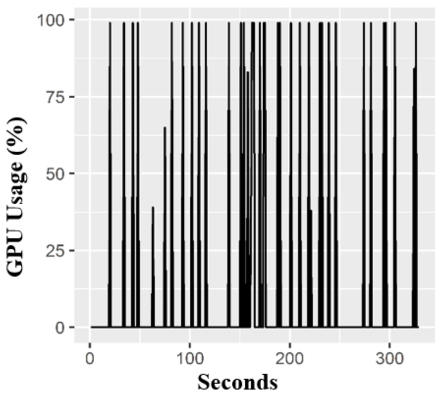

Figure 10. Context-Awareness Module GPU usage graph (%)

6 Conclusion

This paper presents an integrated mobile robotic system that runs multiple vision-based components in real-time. The proposed system implements monocular SLAM and contextual understanding of a scene, which creates a 2D spatial map with detected obstacles. This system showcases the importance of a modular framework which can include latest SLAM and Context-Awareness algorithms in a plug and play format. This system is effective and can be run in real-time on multiple embedded platforms that are integrated as a system. The proposed system is a step forward in making intelligent and contextually aware robots ubiquitous. The results also demonstrate the potential for enabling a computer vision system for autonomous navigation.

Some of the possible extensions and improvements to this projects are documented as follows. For instance, the proposed system does not have an effective way to deal with a large area to be mapped in real-time. A potential solution is to remove older parts of the global map and will be investigated in the authors' future work.

The use of higher resolution input images with more features will result in better tracking and mapping. To ensure that the algorithm still runs in real-time, the feature extraction and matching parts can be moved to the GPU. This will allow us to get better results in a dynamic environment.

The computational load put on each Jetson board shows that the module corresponding to the segmentation task runs heavily on the GPU. The size of the model and high memory usage along with the need for real-time performance restricts the speed of the Husky [39]. Improving the segmentation model in order to reduce the computational load in the Context Awareness Module will address this issue.

References

[1] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," in Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, vol. 1, pp. 534–539, IEEE, 2002.

[2] J. J. Leonard and H. F. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in Intelligent Robots and Systems' 91.'Intelligence for Mechanical Systems, Proceedings IROS'91. IEEE/RSJ International Workshop on, pp. 1442–1447, Ieee, 1991.

[3] K. K. Han and M. Golparvar-Fard, "Potential of big visual data and building information modeling for construction performance analytics: An exploratory study," Automation in Construction, vol. 73, pp. 184–198, 2017.

[4] K. Asadi and K. Han, "Real-time image-to-bim registration using perspective alignment for automated construction monitoring," in Construction Research Congress 2018, pp. 388–397, 2018.

[5] K. A. Boroujeni and K. Han, "Perspective-based image-to-bim alignment for automated visual data collection and construction performance monitoring," in Computing in Civil Engineering 2017, pp. 171–178.

[6] S. Siebert and J. Teizer, "Mobile 3d mapping for surveying earthwork projects using an unmanned aerial vehicle (uav) system," Automation in Construction, vol. 41, no. 0, pp. 1 – 14, 2014.

[7] C. Kropp, C. Koch, and M. KÃűnig, "Interior construction state recognition with 4d bim registered image sequences," Automation in Construction, vol. 86, pp. 11 – 32, 2018.

[8] I. Jeelani, A. Albert, R. Azevedo, and E. J. Jaselskis, "Development and testing of a personalized hazard-recognition training intervention," Journal of Construction Engineering and Management, vol. 143, no. 5, p. 04016120, 2016.

[9] J. Betthauser, D. Benavides, J. Schornick, N. O'Hara, J. Patel, J. Cole, and E. Lobaton, "Wolfbot: A distributed mobile sensing platform for research and education," in American Society for Engineering Education (ASEE Zone 1), 2014 Zone 1 Conference of the, pp. 1–8, IEEE, 2014.

[10] S. Wilson, R. Gameros, M. Sheely, M. Lin, K. Dover, R. Gevorkyan, M. Haberland, A. Bertozzi, and S. Berman, "Pheeno, a versatile swarm robotic research and education platform," IEEE Robotics and Automation Letters, vol. 1, no. 2, pp. 884–891, 2016.

[11] M. Gholizadeh, A. Yazdizadeh, and H. Mohammad-Bagherpour, "Fault detection and identification using combination of ekf and neuro-fuzzy network applied to a chemical process (cstr)," Pattern Analysis and Applications, pp. 1–15, 2017.

[12] NVIDIA, "Unleash your potential with the jetson tx1 developer kit." On-line: https://developer.nvidia.com/embedded/buy/jetson-tx1-devkit, Accessed: 29/11/2017.

[13] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," IEEE robotics & automation magazine, vol. 13, no. 2, pp. 99–110, 2006.

[14] D. Fontanelli, L. Ricciato, and S. Soatto, "A fast ransac-based registration algorithm for accurate localization in unknown environments using lidar measurements," in Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on, pp. 597–602, IEEE, 2007.

[15] J. H. Gao and L.-S. Peh, "A smartphone-based laser distance sensor for outdoor environments," in Robotics and Automation (ICRA), 2016 IEEE International Conference on, pp. 2922–2929, IEEE, 2016.

[16] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," IEEE Transactions on Robotics, vol. 31, no. 5, pp. 1147–1163, 2015.

[17] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017.

[18] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in European Conference on Computer Vision, pp. 834–849, Springer, 2014.

[19] R. Mur-Artal and J. D. Tardós, "Orb-slam2: an open-source slam system for monocular, stereo and rgb-d cameras," arXiv preprint arXiv:1610.06475, 2016.

[20] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp. 15–22, IEEE, 2014.

[21] A. Concha and J. Civera, "Dpptam: Dense piecewise planar tracking and mapping from a monocular sequence," in Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pp. 5686–5693, IEEE, 2015.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587, 2014.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in Advances in neural information processing systems, pp. 91–99, 2015.

[25] R. Girshick, "Fast r-cnn," in Proceedings of the IEEE international conference on computer vision, pp. 1440–1448, 2015.

[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440, 2015.

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1–9, 2015.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, pp. 1097–1105, 2012.

[29] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for scene segmentation," IEEE transactions on pattern analysis and machine intelligence, 2017.

[30] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[31] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," arXiv preprint arXiv:1606.02147, 2016.

[32] Google, "Ros kinetic." On-line: http://wiki.ros.org/kinetic/Installation/Ubuntu, Accessed: 29/11/2017.

[33] ClearPathRobotics, "husky-unmanned-ground-vehicle-robot." On-line: https://www.clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/, Accessed: 29/11/2017.

[34] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," JOSA A, vol. 4, no. 4, pp. 629–642, 1987.

[35] K. Asadi, "Orb slam tracking state." http://goo.gl/4EhtTE, 2017.

[36] K. Asadi, "First demo: Scene 1." http://goo.gl/yxnwY1, 2017.

[37] K. Asadi, "Second demo: Construction site." http://goo.gl/PWeXby, 2017.

[38] K. Asadi, "Third demo: Scene 2." http://goo.gl/ZbofdK, 2017.

[39] K. Asadi, P. Chen, K. Han, T. Wu, and E. Lobaton, Real-time scene segmentation for autonomous robots on construction sites. Computing in Civil Engineering, 2018.