Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 38 (2019) 1524–1531

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2019), June 24-28, 2019, Limerick, Ireland.

Autonomous Navigation of mobile robots in factory environment

Suman Harapanahalli[a], Niall O Mahony[a], Gustavo Velasco Hernandez[a], Sean Campbell[a], Daniel Riordan[a], Joseph Walsh[a]

[a] *Lero – the Irish Software Research Centre, Department of Agricultural and Manufacturing Engineering, Institute of Technology, Tralee, Co. Kerry, Ireland*

Abstract

Navigating through unstructured environments is a basic capability of intelligent creatures, and thus is of fundamental interest in this research. Navigation is a complex task that relies on developing an internal representation of space, grounded by recognizable landmarks and robust visual processing, that can simultaneously support continuous self-localization ("I am here") and a representation of the goal ("I am going there"). Recent advancements in Artificial Intelligence (AI) and related technologies can make this achievable. The number of robots deployed in the manufacturing industry has increased rapidly and this trend is likely to continue in the future, as autonomous robots have the potential to automate a wide array of labor-intensive tasks in the factory environment and improve output. There are many technical challenges that need to be solved to realize an autonomous multifunctional robotic platform. In this research, we aim to address the primary problem of the autonomous navigation of robots in the factory environment. The robotic platform will be able to recognize the markers on the factory floor and navigate in the factory on the designated path by avoiding obstacles in its path from point A to point B autonomously. In this research, we use a minimal number of sensors to reduce the BOM cost of the robotic platform and maximize battery life. We intend to use cameras (RGB), motor encoders and a low-cost IMU to localize the robot, and an electric drive train to propel the platform. Also, we have used neural networks to recognize the markers and paths in the factory environment, Simultaneous Localization and Mapping (SLAM) to localize the robot and a navigation algorithm to guide the robotic platform to the destination.

Keywords: Autonomous unmanned ground vehicle, Stereo vision for autonomous navigation, AI in Manufacturing, Autonomous machines

1. Introduction

Robotics has helped humans greatly in automating many activities in the factory. Robots have revolutionized multiple industrial processes to mass produce products several decades ago, but these robots are hardly more than versatile machines running a complex, but fixed program. In general, manufacturing robots do not exhibit autonomous intelligence. Except for basic control flow, they are mostly unaware of their environment and limited sensory input is used during operation. Therefore, great care is taken so that the environment of the robot is as predictable as possible, e.g., by using fences to lock uncertainties out of the robot's workspace. This lack of autonomy is one of the major obstacles which must be overcome to allow robots to become mobile in a prior known environment. In many cases, robots are controlled manually to move from source to destination. However, several studies have been carried out on autonomous robots leading to a whole panoply of potential applications. Autonomous navigation of robots on the factory floor will bootstrap its capabilities, navigation is a complex task that relies on developing an internal representation of space, grounded by recognizable landmarks and robust visual processing, that can simultaneously support continuous self-localization and a representation of the destination.

When a robot is deployed in an environment such as a factory, it is usually not feasible to equip the robot with an accurate model of that environment in prior. Therefore, the robot will first need to create a model of its world. For a mobile robot, this model generally needs to comprises a map that allows it to localize itself and plan a collision-free path according to its assignment.

2. Motivation

With the advent of Industry 4.0, factory floors will be more connected, with decentralized communications and decision-making being performed on the factory floor. Autonomous robots and smart assembly lines shall become an integral part of the smart factory. Autonomous robots can deliver parts from the storage area and the workstation and load or unload goods from vehicles. In this research, we are attempting to teach autonomous machines about industry environment by adding an industry adaptation layer to the general architecture of an autonomous robotic platform (henceforth referred to as the rover).

Many of the deep learning techniques which are used for robot perception and continuously adapting these technologies to the factory floor would help us produce better quality products at an economical price. Adapting this continuously could become tedious and costly, so we propose a new module in the general autonomous vehicle architecture with which we can augment data at the sensor output to ensure the mobile robot follows its environment rules.

While several autonomous vehicles and robots rely on complex sensors, the following research questions arise:

- Is it possible to build a low-cost AUGV [Autonomous Unmanned Ground Vehicle] ?
- Can the AUGV exhibit reliable autonomous decision making to move from one place to another by understanding its environment?
- Can the AUGV detect obstacles and process this information to calculate the best path to reach its destination?
- Can the system easily adapt to new environments/conditions?

3. Methodology

To answer the above research questions, the methodology followed during this research work has consisted of the following phases:

- Literature Review: An in-depth literature review has been carried out to understand the techniques used in the design of autonomous vehicles.

- Critical Analysis: Different sensors have been evaluated to sense rovers' surroundings environment, different techniques for objection detection and instance segmentation have been studied and obstacle avoidance has also been performed.
- System Architecture Design: We have chosen to use the ROS (Robotic Operating System)[1] framework as the software base for the development of the rover. There are many open source software packages available in ROS ecosystem, and using these resources, will accelerate the development of rover. The new adaptor module can be easily trained to fit into a new environment.
- Implementation and Testing: A prototype shall be developed and system undergoes rigorous testing.

The shall be able to create a map of the occupied and free space, using this shall be able to choose the path to navigate from its current position to the destination. However, in an industrial environment, there are free spaces that should not be used by machinery such as walkways for personnel. The rover should be able to identify space that it can and cannot use and using this information for the navigation route planning shall make the rover fit into the factory environment.

To perform SLAM, we have evaluated the use of 2D LIDAR (Light Detection and Ranging) and found that, in many cases, the object and its orientation are missed in sensing and this has a major impact on the path planning for the rover. For example, consider a case where a forklift is lifting a load, the 2D LIDAR may fail to scan the lifted load. This setup results in a huge blind spot (vertically) in the rover's field of view. One solution to address this could be to use a tilt platform over which the 2D LIDAR is placed, converting data from 2D LIDAR to a 3D LIDAR format and reducing the error could be considered as a different research problem altogether.

There are numerous types of sensors used to sense surrounding environment in 3D, sensors such as 3D LIDAR, Time-of-Flight (TOF) camera and stereo cameras. Microsoft's Kinect and Intel realsense are popular structured-light emitting sensors with stereo camera often used, these sensors project known patterns of light onto the environment to estimate scene depth information. TOF cameras have a better sensing range and higher frame rate but have low resolution. LIDAR sensors have high accuracy and can be operated in outdoor environments in direct sunlight but the accuracy of this degrades in rain and snow. The 3D LIDAR sensors are very high priced this would have a big impact on the BOM cost of the rover. An alternative is to use stereo camera to sense its surrounding environment. More than one image from different angles is required to reconstruct a disparity map of the scene. A disparity map can be converted into a point cloud data using camera parameters. Generated point cloud data can be used for path planning and obstacle avoidance.

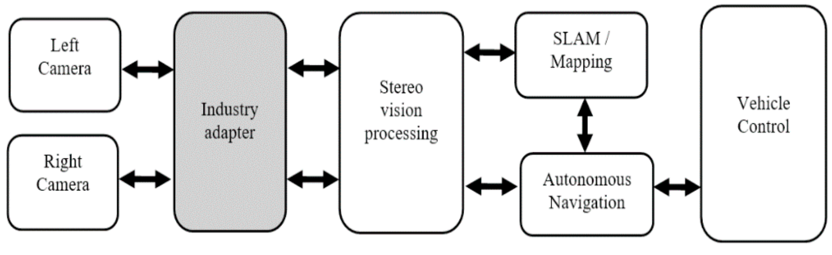

Figure 1:Block diagram of the system

A simplified block diagram of the proposed system is shared figure 1, the industrial adapter module is a new module that we introduce in this research, this module shall be able to recognize the objects and augment data for further processing with SLAM.

The SLAM is feed with augmented data from Industrial adaptor. With object augmentation in non-usable spaces, SLAM will be able to use only allowed space while generating the occupancy map. Using the Industry adaptor as an additional layer the system will be able to adapt in new conditions easily.

3.1. Robotic operating System

Robotic Operating System (ROS) is an open-source, meta-operating system for robotic control. It provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes and package management.

3.2. Stereo Vision for perception

Depth maps from stereo vision can be created using epipolar geometry. Figure 2 below shows a basic setup with two cameras (0 and 0') taking an image of the same scene.

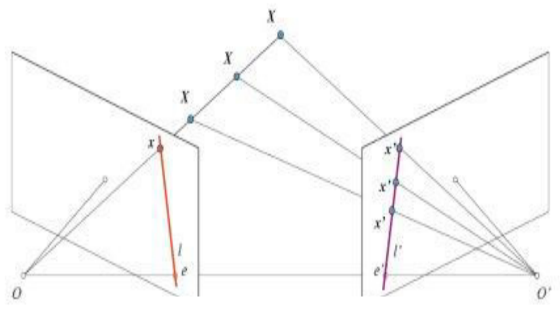

Figure 2: Epipolar geometry

By using only, the left camera, we can't find the 3D point corresponding to the point x in the image because every point on the line OX projects to the same point on the image plane. By considering image from the right camera also now different points on the line OX projects to different points (x′) in the right plane. So, with images from the right and left camera we can triangulate the correct 3D point. There are a few open source packages such as OpenCV to perform triangulation and generate point cloud data[2].

We can also use Cuda-based stereo matching algorithms[3] or a neural network [4] to predict depth from images. The predictions from neural network may tend to fail when encountering new objects or new poses. In this research we have used "usb_cam" [5] to capture images from left and right cameras and for stereo processing we use "Stereo Image Proc" [6].

3.3. Industry adaptor

The industry adaptor is new module introduced in this research, activities of this module are:

- Object recognition – This shall be able to recognize different symbols and their boundaries in the industry.
- Augment virtual objects – Based on the detected objects, this module shall augment objects over ROI.

For Object recognition, we have used neural network-based methods to detect objects of interest from the camera feed. Object augmentation module shall get details of pixel that needs to be augments from object detection module, with this information objects are augmented over the region of interest to make system visualize this space as not free/usable space for rover ex - consider the use case where factory floor has markings for human movement and machinery movement, the rover should not come into the human lane and should always use machinery moment lane only.

We have chosen mask R-CNN network for object recognition and instance segmentation [8], we use Python and OpenCV for objects augmentation. Our object detection neural network can be trained to understand all symbols used in industry like ISO 6790 and other standards, but currently, we have trained our network to detect the human lane.

We can use adapter in multiple ways to describe below –

- Continuous argumentation: Images from camera will be augmented continuously and feed to the localization system

- Mapping argumentation: In this method we will argument images only during mapping process of SLAM and the non-usable spaces will be marked as occupied space.

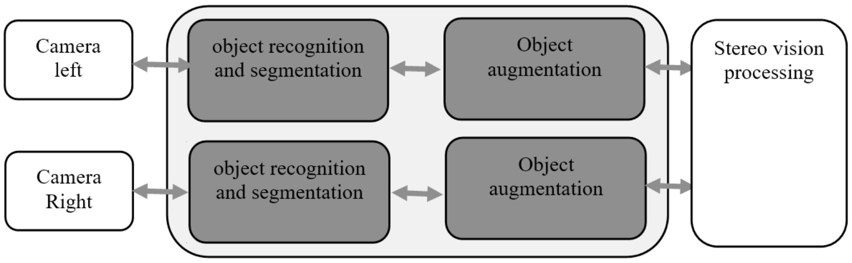

Figure 3: Block diagram for Industry adaptor

In Figure 4(a) the image taken by the camera indicated where the lanes for humans and machinery moments are indicated clear. Figure 4(b) shows image after object augmentation on human lane.

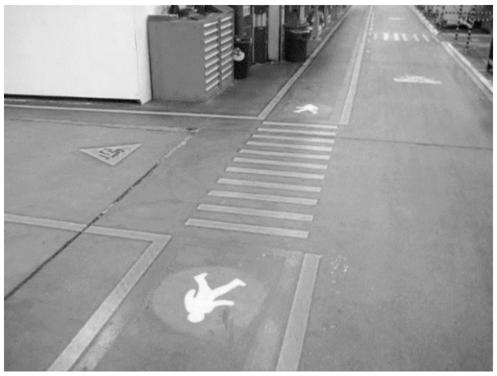

Figure 4(a): Camera image

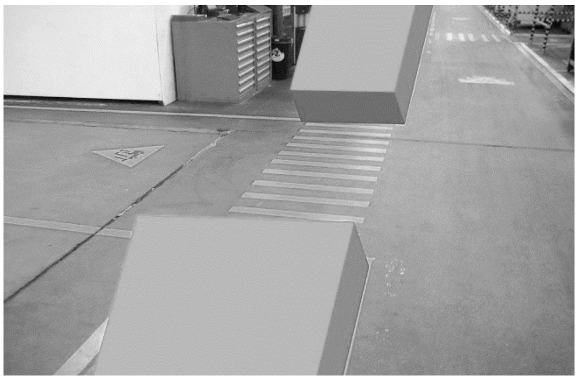

Figure 4(b): Augmented objects over camera image

3.4. SLAM

SLAM refers to the problem of trying to simultaneously localize (i.e. find the position/orientation) with respect to its surroundings, while at the same time mapping the structure of the environment. SLAM is a concept which solves a very important problem in mobile robotics. SLAM is not necessarily a computer vision problem and need not involve visual information at all. It can be done with LIDAR and IMU alone, however here we have considered visual SLAM. SLAM is made up of two parts:-

- Mapping: Building a map of the environment which the robot is in.
- Localization: Navigating in its environment using map while keeping track of the robot's relative position and orientation.

There numerous open source ROS packages available for SLAM like ORB SLAM [9] and there are few integrated solutions which use a stereo camera and perform visual SLAM onboard like the Intel tracking camera T265 [7].

3.5. Sensors for localization

We can broadly classify localization sensors into 2 categories: external source and internal source-based sensors. Internal sensors use rover's velocity and/or wheel motion to localize itself whereas external sensors use RF source for localization. The most popular external sensors are GNSS sensors [ex-GPS], however, performance of these will be degraded by weather conditions and in covered areas. The accuracy of the GNSS sensors could vary between 3 meters and above, there are few GNSS devices with the accuracy of 10 mm but they require a precisely located base station and cannot work in indoor/sheltered environments. Other possibilities include using RF signals from WIFI [8] or Bluetooth device [9] to get rovers relevance position from signal source to localization itself.

Onboard sensors like Inertia Measurement Units [IMU] and motor encoders are common internal sensors. IMU's works by sensing motion including the type, rate and direction of that motion using a combination of accelerometers and gyroscopes. Accelerometers are placed such that their measuring axes are orthogonal to each other [10]. An IMU works by detecting the current rate of acceleration, as well as its changes in rotational attributes, including pitch, roll and yaw. This information is used by the rover to generate its motion trajectories and other purposes.

A motor encoder is an electromechanical device that provides an electrical signal that is used for speed and/or position control. Encoders turn mechanical motion into an electrical signal these are used by the rover to monitor distance travelled, we have encoders on all wheels of the rover to understand its direction and speed of motion.

3.6. Autonomous Navigation

The goal of navigation is to move the rover from a given point to destination by avoiding all obstacles in its path. The simplest algorithm for autonomous navigation is explained below –

- The map is created using a grid or mesh-based representation.
- Each cell in the grid is representing space as free[f], occupied[o] or unknown[u].
- Rover shall be able to use all free cells in the grid for navigation.
- Rover shall be aware of its current position/cell in the grid.
- Given a destination to navigate the rover shall use the shortest path algorithm (Dijkstra algorithm) or A* [11] to calculate the least cost (free cells path) to reach the destination.

The figure 5 gives the representation of the above explanation and rover position is represented as R and destination is represented as "D". The augmented objects from the industrial adaptor are represented as "o" as these are considered as static obstacles.

o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
o	f	R	f	f	f	f	f	f	f	f	f	f	f	f	f	o	o	o	o	o	o	o	o	f	f	f	f	f	f	f	f	o	o	o	o	u	u	o
o	f	f	f	o	o	o	o	o	o	o	o	f	f	f	f	o	u	u	u	u	u	u	o	f	f	f	f	f	f	f	f	o	o	o	o	u	u	o
o	f	f	f	o	o	o	o	o	o	o	o	f	f	f	f	o	o	o	o	o	o	o	o	f	f	o	o	o	o	f	f	o	o	o	o	u	u	o
o	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	o	o	o	o	f	f	f	f	f	f	f	f	o
o	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	o	o	o	o	f	f	D	f	f	f	f	f	o
o	u	u	f	f	f	f	f	f	f	o	o	o	o	o	o	o	o	o	o	o	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	o
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

Figure 5: Occupancy grid map

The complexity of this increases when considering a dynamic obstacle, i.e.; dynamic changes in the free and occupied cells. To manage this, we will have a global planner which will consider all the static obstacles and the local planner to avoid the dynamic obstacles and command velocity (angular and linear) information for the rovers drive train. In a smart factory, global map can be shared with multiple rovers on the factory floor to update its current position and its trajectory to the destination, this will help to have better coordination between rovers and have less impact on the planned path. In this research, we will not consider the multiple rovers and its impact on the planning as it is considered as a separate research problem where collaborative navigation can be addressed [12].

If we consider grid cells to be very small, occupancy grid map will generate a clear picture of free spaces, but the number of cells required to navigate rover will vary depend on the rover size, if the cells are big then we need to have a propagation cost associated with each cell to represent the amount of free and occupied space with the cell.

The Navigation Stack receives data from IMU and motor encoders to generate rover trajectory and odometry information. The sensor transform is required to understand the rovers pose and position with respect to sensor position on rover. A high-level overview of the navigation module and its interaction with other components are shown in the Figure 6.

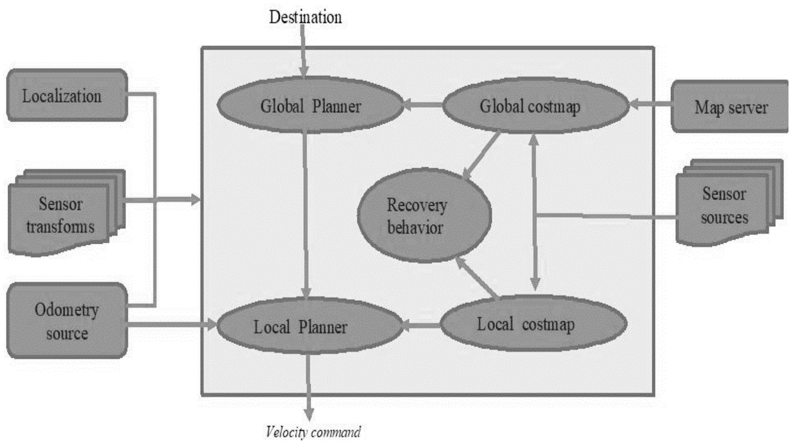

Figure 6: High level overview of the navigation

The cost map uses information from the static map and sensor data to generate an occupancy grid. Sensors are used to either insert obstacle information into the cost map or remove obstacle information from the cost map. An insertion operation is just an index into an array to change the cost of a cell. A clearing operation, however, consists of ray tracing through a grid from the origin of the sensor outwards for each observation reported. The three-dimensional structure of the obstacle is projecting down stored as 2-D and update to the cost map.

A grid-based planner is used by global path planner. The local path planner creates a kinematic trajectory based on the global planner with which the rover to can get from a starting point to destination. The local planner shall also have a controller's which will use trajectory to determine linear (dx, dy) and angular (d-theta) velocities that need to be sent to the rover drivetrain. During navigation rover may get into a blocked position, we need to have means to recover rover from its blocked position.

We are using ROS “Move Base” package [13] for navigation, this is one of the industry standard package used by many unmanned ground vehicles.

3.7. Vehicle Interface

We are using “Differential drive Controller” ROS package to control rovers drive train and split steering. Odometry is determined from motor encoders and shared with SLAM for further usage.

3.8. Results and future work

In this research, an Autonomous Unmanned Ground Vehicle (AUGV) has been developed using open source ROS packages. The rover is feed with the destination from the remote terminal, using this rover starts to compute shortest path to its destination. It uses the stereo camera for its perception, IMU and wheel encoders to localize itself. Velocity commands from the local planner shall control the rovers drive train and steering.

We have recently noticed many open source projects from major like NVIDIA’s Jetbot [14], which use min sensors to perform autonomous navigation, Nvidia provides map editors to define unusable places in its environment. We have used OpenCV to augment obstacles, in future we can use AR core [15] or other SDK to generate obstacles. We have been looking at other means to avoid spaces, map editors are another easier approach to mark unusable spaces. In future we can also develop means to update the maps directly instead of augmenting obstacles.

Acknowledgements

This work was supported with the financial support of the Science Foundation Ireland grant 13/RC/2094 and co-funded under the European Regional Development Fund through the Southern & Eastern Regional Operational Programme to Lero - the Irish Software Research Centre (www.lero.ie).

Reference:

[1] ROS.org | Powering the world's robots, https://www.ros.org/ (accessed 21 February 2019).

[2] OpenCV: Epipolar Geometry, https://docs.opencv.org/3.4/da/de9/tutorial_py_epipolar_geometry.html (accessed 20 February 2019).

[3] Ivanavičius A, Simonavičius H, Gelšvartas J, et al. Real-time CUDA-based stereo matching using Cyclops2 algorithm. *EURASIP J Image Video Process* 2018; 2018: 12.

[4] Smolyanskiy N, Kamenev A, Nvidia SB. *On the Importance of Stereo for Accurate Depth Estimation: An Efficient Semi-Supervised Deep Neural Network Approach*, https://youtu.be/0FPQdVOYoAU. (accessed 20 February 2019).

[5] usb_cam - ROS Wiki, http://wiki.ros.org/usb_cam (accessed 26 February 2019).

[6] stereo_image_proc - ROS Wiki, http://wiki.ros.org/stereo_image_proc (accessed 20 February 2019).

[7] Tracking camera T265 - Intel RealSense Depth &amp; Tracking Cameras, https://realsense.intel.com/tracking-camera/ (accessed 20 February 2019).

[8] Xiang C, Zhang Z, Zhang S, et al. *Robust Sub-meter Level Indoor Localization-A Logistic Regression Approach*, https://arxiv.org/pdf/1902.06226.pdf (accessed 26 February 2019).

[9] Bluetooth 5.1 adds device direction support with location accuracy up to centimetre level | Computing, https://www.computing.co.uk/ctg/news/3070175/bluetooth-51-adds-device-direction-support-with-location-accuracy-up-to-centimetre-level (accessed 26 February 2019).

[10] Hazry D, Sofian M, Zul Azfar A. *Study of Inertial Measurement Unit Sensor*, https://pdfs.semanticscholar.org/4759/77be479ac99a996e7304a95889e24a06a4a1.pdf (2009, accessed 26 February 2019).

[11] Introduction to A*, http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html (accessed 14 May 2019).

[12] Morales JJ, Khalife J, Kassas ZM. *Collaborative Autonomous Vehicles with Signals of Opportunity Aided Inertial Navigation Systems*, https://pdfs.semanticscholar.org/d6bf/67d591e73412ef19a84844afa9b95415bdbe.pdf (accessed 20 February 2019).

[13] move_base - ROS Wiki, http://wiki.ros.org/move_base (accessed 20 February 2019).

[14] BRIAN CAULFIELD. JetBot DIY Autonomous Robot Impresses at GTC | NVIDIA Blog, https://blogs.nvidia.com/blog/2019/03/26/jetbot-diy-autonomous-robot/ (accessed 14 May 2019).

[15] ARCore Overview | ARCore | Google Developers, https://developers.google.com/ar/discover/.