Simultaneous Localization and Mapping for Autonomous Robot Navigation in a Dynamic Noisy Environment.

By

Agunbiade Olusanya Yinka

Student number: 216165423

A thesis submitted in fulfilment of the degree

DOCTOR OF TECHNOLOGY: INFORMATION SYSTEMS

FACULTY OF APPLIED AND COMPUTER SCIENCE

Department of Information Technology,

VAAL UNIVERSITY OF TECHNOLOGY

Promoter: Professor. T. Zuva

November, 2019

DECLARATION AND COPYRIGHT

I hereby declare that this thesis is written by Agunbiade Olusanya Yinka for the fulfilment of D.Tech: Information Technology, and has not been submitted to any institution of Higher Learning apart from VAAL University of Technology. I certainly acknowledge that all sources cited are documented by means of a comprehensive list of references

Agunbiade. O. Yinka

DEDICATION

This thesis is dedicated to God Almighty, the creator and the finisher of my faith. I thank God for his love, care and protection over my life. This thesis is also dedicated to my late mum, Mrs Omoyeni Agunbiade for her unimaginable love, financial support and raising me to be the person I am today.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratefulness to God Almighty, who has been my provider and protector since I was born.

I would love to recognize with special appreciation, the effort of my supervisor: Prof. Tranos Zuva for his inspiration, leadership, support, acceptance and attention during my Doctoral study.

Beyond my supervisors, I would love to appreciate my father Agunbiade Oyebode Jegede for his unlimited care and provision. I will continually live to recall your legacy. I would also love to thank my siblings for their financial support and encouragement. I love you all.

I sincerely appreciate my wife, Agunbiade Temitope Khadijat and my son Agunbiade Oluwapamilerinayo for their support and cooperation, without them I am nothing. I pray that Almighty God grant all your desire in Jesus' name. Amen.

I would also like to thank my friend, Dr Awosejo Johnson, who is always there for me whenever I need advice.

Furthermore, my deepest gratitude goes to the Council for Scientific and Industrial Research (CSIR) for their financial support. Presently, CSIR has invested over $30,000 into this study and without their involvement, the experimental evaluation will be impossible.

Lastly, my gratitude goes to VAAL University of Technology for paying my tuition fees.

ABSTRACT

Simultaneous Localization and Mapping (SLAM) is a significant problem that has been extensively researched in robotics. Its contribution to autonomous robot navigation has attracted researchers towards focusing on this area. In the past, various techniques have been proposed to address SLAM problem with remarkable achievements but there are several factors having the capability to degrade the effectiveness of SLAM technique. These factors include environmental noises (light intensity and shadow), dynamic environment, kidnap robot and computational cost. These problems create inconsistency that can lead to erroneous results in implementation. In the attempt of addressing these problems, a novel SLAM technique Known as DIK-SLAM was proposed.

The DIK-SLAM is a SLAM technique upgraded with filtering algorithms and several re-modifications of Monte-Carlo algorithm to increase its robustness while taking into consideration the computational complexity. The morphological technique and Normalized Differences Index (NDI) are filters introduced to the novel technique to overcome shadow. The dark channel model and specular-to-diffuse are filters introduced to overcome light intensity. These filters are operating in parallel since the computational cost is a concern. The re-modified Monte-Carlo algorithm based on initial localization and grid map technique was introduced to overcome the issue of kidnap problem and dynamic environment respectively.

In this study, publicly available dataset (TUM-RGBD) and a privately generated dataset from of a university in South Africa were employed for evaluation of the filtering algorithms. Experiments were carried out using Matlab simulation and were evaluated using quantitative and qualitative methods. Experimental results obtained showed an improved performance of DIK-SLAM when compared with the original Monte Carlo algorithm and another available SLAM technique in literature. The DIK-SLAM algorithm discussed in this study has the potential of improving autonomous robot navigation, path planning, and exploration while it reduces robot accident rates and human injuries.

Keywords: Autonomous robot, Trajectory, Navigation, Filtering Algorithm, Kidnap problem, Dynamic Environment, Simultaneous Localization and Mapping.

Contents

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

SLAM: Simultaneous Localization and Mapping.

DIK-SLAM: Dynamic, Illumination variation and Kidnapping Simultaneous Localization and Mapping

BatSLAM: Bat Simultaneous Localization and Mapping

EKF: Extended Kalman Filter

PP: Parallel Pipelined

BS: Binary Swapping

EIF: Extended Information Filter

RUKF: Recursive Unscented Kalman Filter

MCL: Monte Carlo Localization

UKF: Unscented Kalman Filter

CEKF: Compressed Extended Kalman Filter

POMDP: Partially Observable Markov Decision Process

FIRM: Feedback-based Information Road Map

DGKD: Double Guarantee Kidnapping Detection.

PDGKD: Probabilistic Double Guarantee Kidnapping Detection

LTKF: Linear Time-Varying Kalman Filter

PTAM: Parallel, Tracking and Mapping

ZMSSD: Zero-Mean Summed Squared Differences

VISP: Visual Serving Platform

RGBD-SLAM: Red, Green, Blue and Depth- Simultaneous Localization and Mapping

3-D: 3 dimensions

2-D: 2 dimensions

KLT: Kanade-Lucas-Tomasi

ICP: Iterative Closet Point

GTSAM: Georgia Tech Smoothing and Mapping

ISAM: Incremental Smoothing and Mapping

ORB SLAM: Oriented fast and Rotated Brief Simultaneous Localization and Mapping

URF: Ultrasonic Range Finder

PIRVS: Percept in Robotics Vision System

CMOS: Complementary Metal Oxide Semiconductor

KVIS: Key-frame Visual Inertial System

VINS: Visual Inertial System

PTAM: Parallel Tracking and Mapping

RANSAC: Random Sample Consensus

PARSAC: Prior-based Adaptive RANSAC

PL-SLAM: Point and Line Simultaneous Localization and Mapping

LSD-SLAM: Large Scale Direct Simultaneous Localization and Mapping

SIP-SLAM: Sparse Interest Point Simultaneous Localization and Mapping

ARM: Advanced RISC Machine

RAM: Random Access Memory

PDF: Proposal Distribution Function

LPF-r: Localization based Particle Filter with Roulette

PF-r: Particle Filtering with Roulette re-sampling

Fast-SLAM: Fast Simultaneous Localization and Mapping

RGB: Red, Green and Blue

HSV: Hue, Saturation and Value

CC: Computational complexity

ATE: Absolute Trajectory Error

RTAB: Real-Time Appearance-Based map

RGB-D SLAM: Red, Green and Blue-Depth Simultaneous Localization and Mapping

RMSE: Root Mean Squared Error

LIST OF PUBLICATIONS

The following publications are produced from this research:

Olusanya Y. Agunbiade, and Tranos Zuva: Simultaneous Location and Mapping in Application to Autonomous Robot. International Conference on Intelligent and Innovative Computing Applications (ICONIC). IEEE Explore Digital Library, DOI: 10.1109/ICONIC.2018.8601094 6-7 Dec. 2018, pp1-5

Olusanya Y. Agunbiade, and Tranos Zuva: Simultaneous Localization and Mapping in a Dynamic Noisy Environment in Real-Time. 3rd World Conference on Smart Trends in System, Security, Security & Sustainability (WorldS4 2019) IEEE Explore Digital Library, pp 1-8

Olusanya Y. Agunbiade, and Tranos Zuva: Simultaneous Localization and Mapping Technique for a Noisy Environment. Published in 4th Interdisciplinary Research and Innovation Conference (IRIC), July 30 - 31, 2019. Vol4. pp 45-50

Olusanya Y. Agunbiade, and Tranos Zuva: Image Enhancement from Illumination Variation to Improve the Performance of Simultaneous Localization and Mapping Technique. This manuscript is submitted to The International Journal of Advanced Manufacturing Technology awaiting a decision.

CHAPTER ONE

1 INTRODUCTION

Autonomous navigation plays an important role that assists robot movement from one point to another without being controlled by anyone and can be achieved by using Simultaneous Localization and Mapping (SLAM) (Montemerlo et al., 2002). These characteristics have attracted researchers and over the years had been extended from a structured environment to other challenging environments such as unstructured terrain and sub-sea terrain with tremendous success (Skrzypczy et al., 2009). Given any known environment, the detailed information will assist in proper planning of future navigation. In an unknown environment, present position estimation is a requirement for building map for an environment. This map will assist in navigation and this represents the general concept of SLAM (Skrzypczy et al., 2009). The problem becomes how to use SLAM to estimate robot position and map for an environment. In SLAM domain, data collected from sensors (camera, laser, sonic and radar) are used to estimate robot position and for map building. This approach tends to be successful because pose estimate of the robot is correlated with feature estimate of the environment (Skrzypczy et al., 2009). In literature, there have been many ways of solving the problem of mapping but the approach can be roughly categorised into features-based and grid-based approaches (Wurm et al., 2003). These techniques have proven to be productive, but the selection is mostly influenced by the nature of the environment. In a large outdoor environment with the predefined object, the feature-based technique is often advised to be used. In clustered and dense surroundings, the grid-based approach offers better performance (Wurm et al., 2003). These two techniques have contributed to SLAM success but they also have their limitations. The grid-based technique suffers from high computational cost and also require huge memory during processing of data but are capable of handling arbitrary objects. The feature-based technique suffers less computational cost but relies heavily on predefined objects. This means some object must be present and must be known, in the absence of these features, the system will fail without recovery (Wurm et al., 2003). Unfortunately, problems associated with SLAM do not have these limitations only but also have others that still need to be discussed. In resolving this issue, general reviews were conducted on various SLAM techniques proposed by past researchers to identify these problems, and this is the reason for why this review study was carried out in chapters two and three. This review covers various SLAM algorithms, their methodology, limitations and advantages. The review results suggested that the computational cost, dynamic environment, illumination variation and kidnap robot are persisting problems limiting the full acceptance of SLAM in the society. This study was conducted on how to address these problems and a novel SLAM technique known as DIK-SLAM was presented. The DIK-SLAM is equipped with multiple algorithms taking into consideration the computational cost as related to the processing speed. Successful implementation has contributed towards improving SLAM research and its acceptance in the industry will improve productivity and safety. This thesis is organized as follows: chapter 2 and 3 discuss the literature review conducted for this research, chapter 4 discusses the methodology. Chapter 5 discusses the experiments carried out and the results obtained while the summary of the study, contribution, conclusion and future work was discussed in chapter 6

1.1 Problem statement

The Simultaneous Localization and Mapping is an important technique as far as autonomous guidance is concerned. Its contribution to successful navigation supports its popularity in the research community. Thus, various SLAM techniques have been proposed with significant breakthroughs, but they are still vulnerable to some problems. Environmental noises (shadow and light intensity) are one of the problems that many researchers complained about. The effect of environment noises makes the analysis of image content difficult to accomplish (Thamrin et al., 2012, Agunbiade et al., 2014). This drawback made some researchers to propose the use of sensors like laser range, sonic and sonar for implementing their SLAM technique (Choi and Maurer, 2016, Byron and Geoffrey, 2013). Thus, environmental noise can also cause kidnapped robot which happens when the robot is not aware of its new position. This arises when a robot moves without noticing its odometry measurement, and the effect can lead to systems failure without recovery (Guyonneau et al., 2012). In the presented research, we minimized the effect of these environmental noises by using filters. Dynamic landmarks (features) were also mentioned as a problem by many researchers in their SLAM technique (Fuentes-Pacheco J, 2015, Clipp et al., 2009) due to their capability to cause biased measurement. The effect leads to map inconsistency and corruption since object position changes due to its dynamic abilities. In our effort to propose a better SLAM technique, all problems were addressed by introducing multiple algorithms for the re-modification purpose taking note of the computational cost. In system enhancement that involves introducing new features, the computation cost is increased. The effect slows down the motion of our system due to a decrease in image processing speed (Hesh and Trawny, 2005). Thus, giving the DIK-SLAM algorithm presented in this study, this problem was minimized by using concurrent operation at the filtering stage. A low computational cost SLAM algorithm based on particle filter was employed to further resolve the effect of response time. However, successful implementation of the DIK-SLAM technique supports better performance than other SLAM techniques proposed in the literature.

1.2 Research questions

Given an Autonomous robot navigating in an environment based on SLAM technique, there is a high possibility of encountering any of the following problems : environmental noises, kidnap problems, computational cost and dynamic environment. The main research question is: How can a novel SLAM technique capable of withstanding the environmental noise in dynamic environment and kidnapping state be developed? In resolving this question, the following sub-questions need to be answered.

- What has been done in the literature to improve on SLAM techniques for autonomous robot navigation?
- How can an improved SLAM technique in comparison with those identified in literature be proposed?
- How can environmental noise and kidnap robot be addressed in a dynamic environment?
- How can the speed of image processing be increased in the proposed SLAM technique?
- What evaluation schemes can be used to evaluate the performance of the proposed SLAM technique?

1.3 Research objectives

The autonomous robot plays an important role that cannot be ignored globally. Based on the main question stated above, the research goal is therefore to improve in real-time the DIK-SLAM technique for perfect navigation of autonomous robots in environmental noises, kidnap problem and dynamic environment scenarios. This could be achieved by executing the following Research Objectives.

- To study and compare SLAM techniques.
- To propose a modified Monte-Carlo algorithm capable of addressing the problem of kidnap robot, environmental noises in a dynamic environment.
- To design a concurrency technique for improving the image processing speed.
- To measure the effectiveness of the proposed SLAM technique.

1.4 Research contributions

The research contribution to knowledge is as follows:

- Development of a novel SLAM technique known as DIK-SLAM for autonomous robot
- Minimizing the effect of environmental noises, dynamic environment and kidnap robot to improve navigations for a robot in their environment.
- Improving the processing speed for the DIK-SLAM techniques
- Advancing robotic research and if deployed in companies and industries using robots for their daily activities, it will increase manufacturing productivities and profit with minimum casualty

1.5 Deliverables

The research deliverables are as follows:

- A novel SLAM technique (DIK-SLAM).
- Published academic papers
- Writing and submission of D.Tech thesis

1.6 Thesis outline

This thesis is structured as follows: Chapter One presents an overview of the entire work. It entails the topic, problem statement, research questions, objectives and contributions of our research. Chapter Two and Three present the literature reviews of related studies .Chapter Four introduces the Methodology of the proposed DIK-SLAM algorithm and the pseudo-codes used in our research. In Chapter Five, presents and discusses various experimental results on the performance of the system. Chapter Six presents a summary of the study, research contributions, research conclusions and future work.

CHAPTER TWO

2 SENSOR, FILTERING, CONCURRENCY AND SLAM ALGORITHM

In an attempt to develop an efficient and effective SLAM technique, sensors are used to acquire the representation of data and sample that play a significant character in object classification but might not be efficient in the presence of environmental noises.

Environmental noises have the ability to degrade the image, making interpretation to be impossible (Agunbiade et al., 2013) and to address this issue, filters can be employed to reduce the presence of noise and improve the quality of the image. This increases the computational cost with an effect of a decrease in processing speed and can be minimised by using concurrency technique (Agunbiade et al., 2013). Meanwhile, SLAM algorithm has the ability to assist in navigation, by means of constructing a map and localizing itself using the same map (Lang et al., 2010). In this chapter, sensors as related to SLAM, filtering algorithms, concurrency technique, dynamic and kidnapping SLAM will be discussed. The discussion covered their advantages, challenges and issues. This chapter has assisted in making an accurate decision of whether to develop or modify or chose an algorithm that will assist the SLAM technique attain its best performance.

2.1 Sensors and SLAM

Simultaneous Localization and Mapping (SLAM) is an important problem that has been broadly researched in robotics. Its support towards the autonomous robot movement has involved scientists in concentrating on this area (Zunino and Christensen, 2001, Fuentes-Pacheco J, 2015). In the past, various techniques for addressing simultaneous localization and mapping has been proposed with remarkable achievements. Research had been conducted based on several types of sensors because of their advantage over another. Sensor selection can be a result of the type of technique a researcher is proposing to solve the problem of SLAM (Steckel and Peremans, 2013). In (Eliazar and Parr, 2003), they proposed to address the SLAM problem in an environment without pre-determined landmarks using laser sensors. The Particle filter algorithm was deployed in their system and had produced a highly detailed map for an office environment. Laser sensors as expensive could produce a wrong measurement when encountering shiny or black objects that do not reflect light, and this could affect robot localization in an environment (Agunbiade et al., 2014). In (Zunino and Christensen, 2001) they proposed the use of the EKF filter algorithm to process the information obtained by the sonar sensors attached to the robot. The sonar sensor was proposed because of its low cost and low computational complexity for retrieving information from the environment. In the work of (Steckel and Peremans, 2013), they condemn the use of sonar sensors because of their inability to provide fine-grained information from the sound. Instead, they proposed bio-sonar. This was employed due to high intelligent interaction capability towards a complex environment and its ability to extract more information from the echoes than sonar. BatSLAM algorithm was proposed to analyse the information acquired by the bio-sonar (Steckel and Peremans, 2013), but from their experiment. The limitation occurs if it encounters a larger complex environment because echoes arriving from different directions are delayed. Trying to analyse them produced an invalid cue which makes the system fail to navigate correctly (Hiryu et al., 2010). In the work of (Irie et al., 2012.), they proposed a vision-based SLAM, because the camera was able to acquire more information from the environment than other sensors. These acquried informations could assist to improve robot navigation (Zehang et al., 2006). Thus, being aware of the issues of environmental noise with a vision-based system, they tend to address the issue of shadow. They proposed the use of two-dimensional occupancy grid maps produced from 3-D point clouds obtained by a stereo camera. They also introduced an extracted salient line segments from the ground into the grid map. On the grid map, robot pose estimation was attained by employing particle filters. In this technique, the grid maps were not affected by shadow and lighting conditions, but under severe illumination conditions. It is impossible to extract the salient line segment which resulted to a failed SLAM technique. However, the issue of illumination variance supports the reason why some researchers still prefer the use of active sensors to acquire data from the environment (Thamrin et al., 2012). Furthermore, the work of (Castellanos et al., 2001) propose the use of multiple sensors for a reliable and accurate measurement of the environment. The idea is for a sensor to take an advantage over the weakness of another. The major limitation is high computational complexity when combining too much data from multiple sensors. Hence, irrespective of the sensor employed, they all still have their limitations. However, sensors are not the only contributor to SLAM failure, the algorithms employed to address the SLAM problem also have their limitations and are discussed extensively in Section 2.2

2.2 SLAM algorithm

Simultaneous Localization and Mapping (SLAM) is a technique that allows a mobile robot to build a map for an unknown environment and simultaneously use the same map to find its location, given a mobile robot navigating through an unknown environment, taking observation of landmarks by using sensors attached to the robot at an instance time K (Dissanayake et al., 2011).. They are used to extract some state variables defined as follows.

x_K represents state vector describing robot orientation and location.

u_K represents control vector, applied at the time $k-1$ to drive the robot to a state x_K at a time K

M_I represents vector describing the location of the i^{th} landmark whose true location is assumed time-invariant

z_I represents observation retrieve from the robot of the location of the i^{th} landmark at a time K

Thus, in multiple observations at any time, these quantities are defined as follows

$x_{I:K} = \{x_1,.......,x_K\}$ represent the history of vehicle locations

$U_{0:K} = \{U_1, U_2,......U_K\} = \{U_{0:K-1}, U_K\}$ represent the history of all control inputs

$M_{1:n} = M_1, M_2,......M_n$ represent set of all landmarks

$Z_{1:K} = \{Z_1, Z_2,....,Z_K\}$ represent set of all landmarks

Given these state variables, they can be used to address the problem of SLAM most especially in static environment. The observation and corresponding particle relating to the trajectory (orientation and location) can be used to generate the map for the environment. However, it must be enhanced to resolve the issue of a dynamic environment and kidnap robot to achieve the study's desirable result.

2.2.1 Dynamic SLAM

SLAM is a technique that was initially implemented for the static environment (Zhang et al., 2011), but there is an environment that is dynamic in nature due to the presence of moving objects. This situation becomes a problem for previous SLAM. In an attempt to address dynamic landmarks, the work of (Oh et al., 2015) proposed an idea of defining dynamic landmarks as an environment that has landmarks location changing, as a result of external forces acting on it. Thus, the change does not affect the information of a robot or other static landmarks. Therefore they proposed an assumption that the dynamic landmarks are independent of static landmarks. In implementing their assumption, they started by proposing the Simultaneous Localization and Mapping problem in a probabilistic form as illustrated in 2-1

$$\rho\left(X_{1:K}, M_{1:n} \middle| Z_{1:K} U_{0:K-1}\right)* \tag{2-1}$$

In an environment that is not only static, the landmarks is re-defined as expressed in equation 2-2

$$M_{1:n} = \left\{M^{S}_{1:n_S},\ M^{d}_{1:n_d}\right\} \tag{2-2}$$

where $M^{S}_{1:n}$ represents numbers static landmarks while $M^{d}_{1:n}$ represents numbers of dynamic landmarks. Based on the assumption, the two landmarks are separated and 2-1 is divided into two parts. An illustration is given below in equation 2-3 and final calculation resulted in equation 2-4

$$\rho\left(X_{1:K}, M_{1:n} \middle| Z_{1:K} U_{0:K-1}\right)*$$

$$\rho\left(X_{1:K}, M_{1:n} \middle| Z_{1:K} U_{0:K-1}\right) \cdot \rho\left(M^{d}_{1:n_d} \middle| X_{1:K}, M^{S}_{1:n_S}, Z_{1:K}, U_{0:K-1}\right) \tag{2-3}$$

$$\rho\left(X_{1:K}, M_{1:n} \middle| Z_{1:K} U_{0:K-1}\right) \cdot \rho\left(M^{d}_{1:n_d} \middle| X_{1:K}, Z_{1:K}\right) \tag{2-4}$$

The equations 2-3 and 2-4 support that Simultaneous Localization and Mapping can deal with static and dynamic landmarks separately and independently. It also supports that dynamic landmarks are independent of each other. Then equation 2-4 is factorized which resulted in equation 2-5

$$\rho\left(X_{1:K}, M_{1:n} \middle| Z_{1:K} U_{0:K-1}\right) \cdot \prod_{i=1}^{M_d} \rho\left(M_{1:n_d}^{d} \middle| X_{1:K}, Z_{1:K}\right) \qquad 2\text{-}5$$

The left part of the equation 2-5 is equal to the equation given in 2-1, which represents the original Simultaneous Localization and Mapping problem represented in a probabilistic form to solve static landmarks only (Oh et al., 2015). However, adding the part of the dynamic features clarify that it is possible to address the problem of Simultaneous Localization and Mapping in a dynamic environment. Thus, this technique will not be adopted because the Extended Kalman Filter (EKF) has a limitation of High computational cost, which the study is taking into consideration (Chen and Lum, 2014). Alternatively, a low computational cost SLAM technique such as a particle-based algorithm will be preferred to EKF.

2.2.2 Kidnap SLAM

The kidnaping problem occurs when the robot moves from one position to another without having any information about its new position (Guyonneau et al., 2012). This can happen as a result of failing sensors or increasing in measurement noise (Negenborn et al., 2003). The effect of this limitation leads to a robot inability to estimate its pose. Therefore, violating the principle of addressing the problem of SLAM, because pose estimation is a requirement for map building and can lead to robot failure without recovery (Guyonneau et al., 2012). Supposing an autonomous robot navigating in an unknown environment with an on-board sensor-based SLAM technique, and the system characteristic of discrete- time dynamic is the equation given in 2-6

$$\begin{cases} x(k+1) = F\left(x(k), u(k)\right) \\ y(k) = g_g\left(x(K)\right) \end{cases} \qquad 2\text{-}6$$

The autonomous robot position $x(k)=(\mathrm{x}_1(k), x_2(k), \theta(k))$ is defined by its location $(x_1(k), x_2(k))$ and its orientation $\theta(k)$ in the environment represented as E at the discrete-time k the function f signifies the robot dynamic and the vector $u(k)$ represents the control vector at a time k. The vector $y(k)=(y_1(k),........,y_n(k))$ signifies a set of vector measurements. Note that $y(k)$ is dependent on $x(k)$ and E. The environment (E) where the movement of the robot takes place is estimated by a grid map. The grid map represented as G is made up of $n\,x\,m$ cells (i, j) and at each cell (i, j) is related $g_{i,j} \in \{0,1\}$. The expression is given in equation 2-7

$$g_{i,j} = \begin{cases} 0 \text{ if the cell corresponds to a free subspace} \\ \text{ of E,} \\ 1 \text{ else} \end{cases} \tag{2-7}$$

where G represents the discrete version of E. Thus, when a robot is kidnapped in an environment, algorithms selected to solve this problem must be able to carry out these objectives: the ability to perform pose estimation, detect kidnap problem and global localization (Guyonneau et al., 2012). However, in the literature, the initializing localization technique has been positive to kidnap robot (Se et al., 2001). In this technique, an extensive search of the current observation over the reference map in the database (pre-mapped environment) will be carried out to find the robot position in relation to this map. This will assist the robot to start up again at the last stop position. However, there is a situation where this technique can experience failure. For instance, if the current observation has changed as a result of dynamic characteristic, the selected reference map might not perfectly match the current observation and could lead to SLAM failure without recovery. Given these reasons, this technique will not be adopted. A different approach towards addressing the problem of kidnapping robot will be proposed in this research.

2.3 Filtering algorithms

Noises are unwanted information that adulterate an image. Their existence in the image come from numerous sources. The image acquisition phase which is the first stage in the SLAM technique is the primary source of how noises appear in digital images.However, there are numerous other ways that images are exposed to noises, depending on how they are created. In vision SLAM technique, environmental noises contaminating images are captured by a digital camera and lead to system inability to understand the image properly. This limitation can be minimised by using a filtering algorithm to reduce the concentration of the noise and this will better improve the image content interpretation. In the literature, various filtering algorithms for noise reduction have been proposed, but it is important to select, improve, or create the best filtering algorithm to be used for the SLAM technique. Section 2.2.1-2.2.3 explains some of the filtering algorithms found in literature.

2.3.1 Bilateral filters

This bilateral filter smoothens images without affecting edges. These filters function based on a non-linear combination of neighboring image values. In this technique of filtering, every pixel is replaced by an average weight of its close pixel. The weight (w_p) assigned, is based on using intensity difference and spatial closeness. Gaussian distribution can be employed to play a significant role for the weight estimation and the result thereafter is a filtered image. The bilateral filtering is expressed in equation 2-8 (Singh and Goyal, 2014).

$$I^{filtered}(x) = \frac{1}{w_p} \sum_{x_i \in} I(x_i) f_r(\|I(x_i) - I(x)\|) g_s(\|x_i - x\|) \; , \qquad \text{2-8}$$

where

$$w_p = \sum_{x_i \in} f_r \left(\left\| I(x_i) - I(x) \right\| \right) g_s \left(\left\| x_i - x \right\| \right),$$

I represents original input image with noise, x denotes current pixel coordinate to be filtered, f_r signifies range kernel for smoothing differences in intensity, g_s represents spatial kernel and the smoothing differences in the coordinate. The g_s and f_r can be a Gaussian function. The algorithm result is impressive and has a wide application different from filtering e.g. tracking and navigation, the major limitation is that input noises still exist after filtering because photometry (intensity) makes neighboring pixel weight very small compared to the pixel of the impulse noise with large weight (Huang and Fuh, 2006).

2.3.2 Wiener filters

Wiener filtering algorithm functions on dark channel technique presented in the study of (Zou et al., 2013). It is a common technique to minimise image noise. Wiener filtering algorithm is linear and shifts invariant filter that has a limitation not suitable for images with edges but can be improved by using the combination of a different method like median filter (Krajsek and Mester, 2004). In this combination technique, edges are preserved because an image restoration problem is converted to an optimization problem. Using the combination technique has a limitation of high computational cost because of transformation and re-transformation in the procedure of implementing this technique. Giving a moving camera with excessive moderate shutter speed, the pixel will be an amalgam of intensity from the point in line with the camera movement as represented in equation 2-9

$$G(u,v) = F(u,v) \cdot H(u,v) \tag{2-9}$$

where F represents the fourier transform version of a given image and H represents the blurring function.

If G and H are known, F can be estimated. F represents the image that has been recovered from noise. The best method to solve the problem of estimating F is wiener filter according to equation 2-10.

$$F(u,v) = |H(u,v)|^2 . \frac{G(u,v)}{|H(u,v)|^2 .\ H(u,v) + K(u,v)} \qquad \text{2-10}$$

where K represents constant selected to improve the estimate.

The algorithm is a widely used technique because of the capability to remove the white area of noise in images and it is easily implemented (Singh and Goyal, 2014).

2.3.3 Median filters

Median filters a popular de-noising and smoothing algorithm. Given a set of variables in random as expressed in equation 2-11.

$$\chi = (Y_1, Y_2, \ldots\ldots\ldots, Y_N) \qquad \text{2-11}$$

The statistic orders expressed in 2-12 are random variables well defined by arranging the values of Y_i in increasing order.

$$Y_{(1)} \le Y_{(2)} \le \ldots\ldots\ldots \le Y_{(N)} \qquad \text{2-12}$$

The median value (χ) using the above equation is presented in equation 2-13.

$$median(\chi) = \begin{cases} Y_m & for\ \ N = 2k+1 \\ \frac{1}{2} Y_k + Y_m & for\ \ N = 2k \end{cases} \qquad \text{2-13}$$

where $m = k+1$ represents the median rank

The median is a good estimator for the location parameter of distribution and has found various applications in filtering and smoothing especially for data corrupted by impulsive noise. Given a gray-scale input image with intensity value $Y_{i,j}$, the 2-D median filter is expressed in equation 2-14

$$x_{ij} = \underset{(r,s)\in w}{median}\left(Y_{i+r,j+s}\right) \qquad 2\text{-}14$$

where w represents window size $m \times m$ where filtering is applied, r, s signifies corrupted pixel in w that needs to be restored (Kirchner and Fridrich, 2010).

This is one of the most common filters but limitation arose when random intensity noise exists in the image, it is less effective and computationally intensive (Aboudaya et al., 2006). Apart from the above-mentioned filters, many more exist like mean shift filters, contrast limited adaptive histogram. We adopted some filtering algorithms different from the ones discussed above, because from the work of the previous researchers, these adopted algorithms attained a reliable result.

2.4 Concurrency technique

This is a form of operation where many computations are carried out simultaneously, based on the principle that, bigger data can be divided into smaller data and be solved concurrently (Parker et al., 2001). These techniques were introduced to overcome the limitations of the serial algorithm. Concurrency operation is used for high-performance computing for many years and has been dominating the area of computer architecture especially in multi-processors. Concurrency (Parallel) operation are much difficult to implement than the sequential algorithm, more especially the communication and synchronization between various sub-tasks are the main issue to achieve a good concurrency operation. Various types of concurrency operations exist but the two commonly used are Parallel Pipelined (PP) and Binary Swapping (BS) (Chin et al., 2001). These two are discussed in sections 2.4.1 and 2.4.2 respectively. In this study, it is important to develop, or to create, or to select the best concurrency operation to reduce high computational cost in the proposed SLAM algorithm.

2.4.1 Binary swapping

Recently, this method became a commonly used technique than others because it allows more simultaneous operations in image composition phase with all the processors busy throughout the communication procedure, but the limitation is that the technique can only be implemented when processors are in a power of two (Lin et al., 2003). In communication and synchronization, the Binary Swapping technique is a divide and conquer algorithm. At first, the input image is divided across processors for every communication step. Processor divides its partial image into two halves. 2 processors are paired for an exchange for half of their partial image, afterward, it is the composition operation. After (K) communication step, each processor held a portion of the final image. The final image will be generated by using the gathering directive to retrieve the portions of partial images from processors holding it (Lin et al., 2003).

2.4.2 Parallel pipelined algorithm

Parallel Pipelined is a common algorithm because it supports arbitrary processor and was designed for the mesh network. In the communication and synchronization using $P = P_r \times P_c$, two stages are involved after the division of the input image across numbers of processors. Row processors (P_r) are arranged in a ring topology and each processor divides its partial image into P_r blocks. Each processor sends a block to the next processor and receives from its previous processor. Processors composite the received block by using over operation. In the second stage, similar to the first operation, column processors (P_c) are arranged in a ring topology and each divides its block into P_c sub-block for a send and receive operation between the next and previous processors respectively. After (K) communication step, the final image can be generated by gathering the sub-blocks from each processor, but the limitation is that some processors are idle and the image composition overhead is high (Lin et al., 2003). Several techniques for concurrency operation exist but the adopted technique (Rotating Tilling) was employed because of its ability to overcome the limitation of Binary Swapping and Parallel Pipelined algorithm.

2.5 Chapter summary

In order to develop a reliable SLAM technique, an extensive review was conducted on the components that can assist to address the problem of SLAM. This chapter discussed the study conducted for the sensor, filtering, concurrency and SLAM techniques in detail and has assisted in selecting the algorithm for developing the proposed SLAM technique

CHAPTER THREE

3 REVIEWS ON SLAM TECHNIQUES

SLAM is an important study drawing the attention of scholars because of its support towards Autonomous Navigation (Se et al., 2001). Autonomous Navigation controls robot movement without human supervision. This becomes a key achievement towards self-exploratory expeditions. Towards achieving proper navigation, numerous SLAM techniques have been proposed over the last 30 years with a remarkable result attained (Lang et al., 2010, Jia et al., 2016. ., Jean-Arcady and David, 2003). This Chapter reviews various SLAM techniques, their advantages and limitations. It also discusses the challenges, open issues and research direction for future SLAM.

3.1 Foundational SLAM Technique

In SLAM technique, sensors play an important role in acquiring data from the environment (Zehang et al., 2006), but the localization and mapping techniques are not limited to this operation. There are procedures that still need to be implemented and the analysis of the data captured by the sensor assists in mapping building and localization. This can be attained by using SLAM algorithm (Chen, 2013). In literature, several foundational SLAM algorithms have been proposed with an outstanding result, but they are all confronted with various challenges and issues (Hadji et al., 2014). In this sub-section, some of these algorithms were discussed together with their limitations and advantages

3.1.1 Extended Kalman Filter

It is important to mention the Kalman Filter because it is the foundation for EKF and some algorithms like Extended Information Filter (EIF), non-linear least-square, etc. (Hadji et al., 2014). Researchers over the years have employed Kalman Filter as an algorithm to estimate dynamic linear systems with Gaussian noise (Chen, 2013). Kalman Filter represents a state vector (μ_t) as illustrated in equation 3-1.

$$\mu_t = (S_t, l_1, l_2 ...., l_N) \qquad 3\text{-}1$$

It is formulated by estimating the landmark (l) where N represents numbers of map landmarks, the current pose (S_t). However, the issue of non-linearity in the robot model is the limitation of the Kalman Filter algorithm and trying to address this issue led to the Extended Kalman filter (Hadji et al., 2014).

Extended Kalman Filter (EKF) is an upgraded version of a Kalman filter that can address the non-linear model (Chen, 2013). The linearization of the non-linear model can be solved by many methods but in EKF, a technique called first-order Taylor expansion is employed to address this issue. At each time (t), it linearizes the measurement and motion model using the current state to estimate for a new update (Chen, 2013). The filter procedure is attained with two steps given in sections 3.1.1.1 and 3.1.1.2.

3.1.1.1 The time update stage

At this stage, the filter computes the covariance matrix $\overset{\wedge}{\sum}_t$ and the predicated state $\overset{\wedge}{\mu_t}$ at the time (t). Expressions is given in equations 3-2 and 3-3.

$$\overset{\wedge}{\mu_t} = f(\mu_{t-1}, u_t) \qquad 3\text{-}2$$

$$\overset{\wedge}{\sum}_t = A_t \sum_{t-1} A_t^T + G \wedge_u G^T \qquad 3\text{-}3$$

where A_t signifies the Jacobian of the motion model f as related to the robot pose that is evaluated at the robot control μ_t, $\wedge_u$ signifies the covariance matrix related to this stage and G represent a projection matrix.

3.1.1.2 The measurement update stage

This stage plays a significant role to address the problem of data association (c) and generates the newly updated measurement for μ_t and $\sum_t$ using the current state of the previous stage. They are computed by estimating first, the Kalman gains as given in equations 3-4 – 3-6.

$$K_t = \hat{\sum}_t C_t^T \left(C_t \hat{\sum}_t C_t^T + \wedge_Z \right)^{-1} \qquad 3\text{-}4$$

$$\mu_t = \hat{\mu}_t + K_t \left(z_t - h\left(\hat{\mu}_t, c \right) \right) \qquad 3\text{-}5$$

$$\sum_t = (I - K_t C_t) \hat{\sum}_t \qquad 3\text{-}6$$

Where I represents identity matrix, K_t represents the Kalman gain, C_t represents the Jacobian of the measurement model h in relation to every detected landmark estimated at $\hat{\mu}$ and the pose of the robot, $\wedge_z$ signifies the covariance matrix as related to this stage and z_t represent the sensor measurement (Chen, 2013).

The two steps listed above are unique to EKF and can be used to instantiate the online SLAM given a condition of the Gaussian model. The EKF is a popular algorithm because of the ability to overcome the problem of the Kalman filter, but the computational cost of the algorithm is high (Dissanayake et al., 2011).

3.1.2 Recursive Unscented Kalman Filter (RUKF)

RUKF is a common algorithm used by the previous researcher to address SLAM problem (Lee et al., 2006, Zanetti, 2012). Its operational functions are similar to that of EKF and KF (Lee et al., 2006). However, EKF and KF suffer limitations and improvements towards these algorithms lead to RUKF and EIF (Hadji et al., 2014). In RUKF, the SLAM posterior probability distribution function was simplified by the assumption that robot position and landmark/features location are independent between a state, which reduces the dimensionality and allows computational complexity of RUKF to be reduced. It is also robust towards non-linearity model than EKF (Lee et al., 2006). Given a SLAM technique that is RUKF based, various functional steps with an independent estimation of robot pose and landmark/features are provided below.

Step 1: The joint Gaussian distribution is estimated for updating recursively the robot position.

Step 2: The marginal probability of the robot position is obtained from step 1

Step 3: step 1 and 2 is repeated for η iteration for obtaining the posterior probability distribution of the robot pose

Step 4: the joint Gaussian distribution is estimated for updating the observed landmark/feature

Step 5: The marginal Gaussian probability distribution of the observed landmark/feature will be obtained from step 4

Step 6: step 4 and 5 are repeated until all observed landmark/ features are updated individually using expression 3-7

$$\approx \eta_i \int \left(P\left(z_i \middle| x_t, m_{C_I}\right) P\left(x_t \middle| Z_t, Z^{t-1}, u^t\right) \times P\left(m_{C_I} \middle| Z^{t-1}, u^{t-1}\right)\right) dx_t \qquad 3\text{-}7$$

where η_i represents the normalizing constant, z_i represents observation measurement, u^t signifies multiple robot control, x_t denotes robot pose at the time t, Z_t signifies multiple simultaneous observation at the time t and m_{C_I} represents observe landmark location.

Thus, the work of (Lee et al., 2006) discovered that the computing time at the feature extraction and matching stage of the RUKF algorithm is still high and should be addressed in future work.

3.1.3 Extended Information Filter

Considering the limitation of the high computational cost of EKF, this shifted the researcher's attention to a more advance filter known as an Extended Information Filter (EIF). In EIF, high computational cost was reduced by avoiding the computation of the Gaussian posterior in terms of the mean μ and the covariance matrix $\sum$. Instead, the filter employs the information form of the posterior, based on the information matrix H and information vector b (Chen, 2013). The parameterization is given below in equations 3-8 and 3-9

$$H = \sum{}^{-1} \qquad 3\text{-}8$$

$$b = \mu^T H \qquad 3\text{-}9$$

The EIF operational function is related to EKF, but the time update and measurement update stage is parallel to that of EKF. The equation 3-5 and 3-6 of the Extended Kalman Filter are transformed in EIF using the generated parametric value of information vector and information matrix (Chen, 2013). The parametric substitution for these stages is given in equations 3-10 – 3-13.

3.1.3.1 The Time Update stage

$$\hat{H}_t = [(I + A_t)H_{t-1}^{-1}(I + A_t)^T + S \wedge_u S^T]^{-1} \qquad 3\text{-}10$$

$$\hat{b} = (b_{t-1}H_{t-1}^{-1} + f(\mu_{t-1}, u_t)^T)\hat{H}_t \qquad 3\text{-}11$$

where I represents identity matrix, A_t signifies the Jacobean of the motion model f estimated at μ_t, the projected matrix is signified as S. At this stage, the updated time estimated is managed to its best because of the mean (μ_t) recovery and the inversion of a dense H_t. However, constant-time updates can be attained if H_t has a sparse characteristic and μ_t is available for all landmarks and robot poses, this is attained at the next stage (Chen, 2013).

3.1.3.2 The measurement updated stage

$$H_t = \hat{H}_t + C_t \wedge_z^{-1} C_t^T \qquad 3\text{-}12$$

$$b_t = \hat{b}_t + \left(z_t - \hat{z}_t + C_t^T \mu_t \right)^T \wedge_z^{-1} C_t^T \qquad 3\text{-}13$$

where $z_t - \hat{z}_t$ represent the difference between updated and current sensor measurement, C_t signifies the Jacobean of the measurement model estimated at μ_t. At this stage, it requires only constant time summation taking into consideration that C is sparse with non-zero values for the observe landmarks and pose in the measurement (Chen, 2013). Given this condition for every step of a matrix, non-zero values will only be associated to $\hat{H}_t$. EIF is regarded as an approximation approach with an improved processing speed better than EKF because of low computational complexity. However, as a result of approximation, the issue of inconsistency arose and has not been fully resolved (Dissanayake et al., 2011).

3.1.4 The Original Monte-Carlo Algorithm (MCL)

This algorithm was presented by (Thrun et al., 2001) and has been modified over the years to create new version and will be discussed in this section. The main prerequisite for a robot in an unfamiliar environment, it's ability to autonomously navigate perfectly without been controlled by anyone and can be achieved by SLAM (Fuentes-Pacheco J, 2015). In Monte Carlo localization-based (MCL) SLAM, it is impossible for the robot to know its exact coordinate and direction in the given map. Thus, the robot needs to extract information from its environment (L et al., 2010). This extracted state is known as belief (Thrun et al., 2001).

In Monte Carlo localization, subsequent belief is represented by a set of samples. An expression is given in equation 3-14 (Bukhori and Ismail, 2017). Samples are a hypothesis for state representation such as wall and obstacle boundaries, are orthogonal,which was proposed by (Jean-Arcady and David, 2003). These sets of samples are used to extract features that will guide the robot trajectory.

$$S_t = \left\{ s_t^{[n]}, w_t^{[n]} \right\}_{n=1,.....,N} \qquad 3\text{-}14$$

where the particle $s_t^{[n]}$ signifies the hypothesis for representing the state pose at time t ,while $w_t^{[n]}$ denoting the weight of the particle indicating how likely the sample represents the state pose.

The basic MCL sequential step is illustrated below fromr steps 1-3, which computes the set of S_t recursively from the previous set S_{t-1}. The MCL technique processes the input of S_{t-1} together with the control state u_t, the sensor measurement z_t and the map M that will be used to generate an output of new particle set S_t at the time t. $\overline{S_t}$ signifies the temporary particle used to represent the belief $\overline{bel}(s_t)$. At each iteration, the temporary particle set $\overline{S_t}$

and the particle sets S_t are empty for newly generated ones. Thus, this recursive procedure continues until the robot reach is the goal and can be realized in three major steps as illustrated below.

Step 1

The equation 3-14 generates new samples $s_t^{[n]}$ based on the previous sample $s_{t-1}^{[n]}$, the control state u_t and the map M . Afterward, is the distribution of the pair $\left(s_t^{[n]}, s_{t-1}^{[n]}\right)$ according to the product distribution as provided in equation 3-15 based on sampling importance resampling algorithm. This distribution is known as proposal distribution.

$$p\left(s_t^{[n]} \setminus s_{t-1}^{[n]}\right) \times bel\left(s_{t-1}^{[n]}\right) \qquad 3\text{-}15$$

Step 2

The importance weight $w_t^{[n]}$ for an individual particle $s_t^{[n]}$ is calculated. this is emphasized to avoid a mismatch between the desired target distribution and proposal distribution as presented in equation 3-16

$$\eta p\left(z_t \setminus s_t^{[n]}\right) p\left(s_t^{[n]} \setminus s_{t-1}^{[n]}, u_t, m\right) bel\left(s_{t-1}^{[n]}\right) \qquad 3\text{-}16$$

Thus, the weight particle set $\overline{S_t}$ represents the posterior belief $bel(s_t)$ that has not been distributed and it will be addressed in step 3 using the resampling technique.

Step 3

This stage is used for carrying out particle re-sampling processes for distributing the weight of a particle set $\overline{S_t}$ according to the posterior beliefs $bel(s_t)$. Given a SLAM technique that is particle-based, using a fixed set of the particle filter in an infinite space might not be efficient.

The effect may cause particles to disperse and eventually lose robot position that will lead to SLAM failure (Milstein, 2008). This situation makes resampling important. In literature, various resampling technique exists but the effective sample size is a useful metric to determine whether resampling is necessary (Carlone et al., 2011), an expression is given in equation 3-17

$$\tilde{N}_{eff} = 1 \Big/ \sum_{n=1}^{s} \left(w_t^{[n]}\right)^2 \qquad \text{3-17}$$

In this technique, particles are resampled if the prior quantity descends less than a given threshold that is fixed at $\frac{n}{2}$ for detail discussion see (Stachniss et al., 2004). The resampling transforms the temporary particle set $\overline{S_t}$ into a new particle S_t. However, before resampling, the temporary particle set $\overline{S_t}$ is distributed according to the posterior belief $\overline{bel}(s_t)$ and thereafter, the particle set is distributed according to $bel(s_t)$. These steps repeat accordingly until the final destination is attained by the robot. These algorithm procedures are implemented at low computational cost and were selected in this study because of this advantage. However, attention will be focused on how this algorithm can be re-modified to address the problem of a dynamic environment, illumination variation and kidnap problem without increasing its computational complexity beyond acceptance.

3.1.5 Rao-Blackwellized particle filter

In the work of Murphy, the Rao-Blackwellized particle filter is implemented to estimate the joint posterior $\left(p\left(S_{1:t}, m \middle| Z_{1:t}, U_{1:t-1}\right)\right)$ of a map (m) and the robot trajectory $\left(S_{1:t} = S_1, \ldots\ldots.., S_t\right)$. The estimation is carried out given the odometry measurement $\left(U_{1:t-1} = U_1, \ldots\ldots.., U_{t-1}\right)$ and the observation measurement $\left(Z_{1:t} = Z_1, \ldots\ldots.., Z_t\right)$ of the robot. Using this information provided, the Rao-Blackweilized particle filter used the factorization given in equation 3-18 to represent SLAM (Wurm et al., 2003).

$$p(S_{1:t}, m | Z_{1:t}, U_{1:t-1}) = p(m | S_{1:t}, Z_{1:t}).p(S_{1:t} | Z_{1:t}, U_{1:t-1}) \qquad 3\text{-}18$$

In the factorization procedure, the position of the robot is first estimated, then the map given that same position is computed. This is mandatory because the map creation relies on the estimated robot pose (Wurm et al., 2003). The Rao-Blackwelized particle filter offers efficient computational cost with improved processing speed. Its representation in equation 3-18 can be formulated efficiently to address SLAM since the posterior of the map $p(m | S_{1:t}, Z_{1:t})$ will be estimated by employing the technique of mapping with known poses, given that $S_{1:t}$ and $Z_{1:t}$ are known (Wurm et al., 2003). In computing the posterior $p(S_{1:t} | Z_{1:t}, U_{1:t-1})$ for potential trajectories, a particle filter may be employed because every particle corresponds to a potential robot trajectory and individual maps are connected with each sample. Therefore, maps will be created using the observation and corresponding particle relating to the trajectory. This procedure allows the robot to learn models of their environment and estimate successfully their trajectory (Wurm et al., 2003). Thus, the Rao-Blackwellizied algorithm and Monte-Carlo algorithm (MCL) are not the only particle-based technique. There are others like FAST-SLAM (Abouzahir et al., 2014), but the particle filter's effectiveness and complexity rely heavily on the number of particles. The increase in the number of particles might improve its effectiveness though at a price of high computational cost. Otherwise, the effectiveness can be minimized with low computational cost. However, estimating an optimal number of particles required is often difficult to attain (Montemerlo and Thrun, 2003).

In the literature, the foundational SLAM algorithm is not only limited to the above-mentioned ones but also on many more others such as Unscented Kalman filter (UKF) and Compressed Extended Kalman Filter (CEKF) (Hadji et al., 2014). These algorithms have better performance over each other but they all have their drawbacks, which have contributed to the limited progress in SLAM research (Dissanayake et al., 2011). However, from the study conducted on the foundation SLAM algorithm, we noticed that these algorithms were able to solve the limitation encountered by each other. For instance, the issue of the non-linearity of the robot model in KF is solved by RUKF and EKF, but the problem of high computational complexity has not been agreeably resolved (Dellaert et al., 2010). This is common among these algorithms and often complained by these researchers (Chen, 2013, Hadji et al., 2014). However, in this research, the computational complexity will be taken into consideration.

3.2 Recent SLAM Technique

Researchers have presented recent SLAM techniques that have been proposed to overcome some of the challenges of the foundational SLAM algorithm, but the SLAM issue has not been fully addressed. Therefore, it is important to conduct reviews on the recent SLAM technique to inform the researcher on the discussion of the methodologies, frameworks and the limitations for their proposed SLAM technique. This has assisted to disclose problems that are persisting to date.

In the work of (Agha-mohammadi et al., 2015), they propose to replan at every time (k) when there is a probability distribution update on the state of the autonomous robot. This technique is referred to as Simultaneous Localization and Planning. The algorithm employed to carry out this task is known as Partially Observable Markov Decision Process (POMDP). This algorithm was proposed because of its ability to cope with uncertainties and changes. The idea employed in motion planning under uncertainty is to identify a policy (π_k) at each time (k) that generates a control state (U_k) using the available information of the robot. Thus, there are other important terms that need to be defined to achieve this goal. Given a robot in an unknown environment whose state is represented by x_k at the time step (k), the motion noise and control state at the time (k) is represented by W_k and U_k respectively. The state evolution model can now be formulated using equation 3-19

$$x_{k+1} = f\left(x_x, U_k, W_k\right) \tag{3-19}$$

However, in any partial observable system, the sensor vector measurement at the very time k represented as Z_k plays an important role in providing observation measurement. The expression Z_k is given in equation 3-20.

$$Z_k = h\left(x_k, V_k\right) \tag{3-20}$$

where V_k denotes sensing noise.

On this note, the data available for deriving decision at each time k is the history of controls and observation as expressed in equation 3-21, the conditional probability distribution for the overall possible robot state is given in equation 3-22.

$$\mathrm{H}_k = \left(Z_{0:K}, U_{0:k-1}\right) = \left(Z_0, Z_1, ..., Z_k, U_0, ..., U_{k-1}\right) \tag{3-21}$$

$$b_k = p\left(x_k \middle| H_k\right) \tag{3-22}$$

The b_k generated in equation 3-22 is also referred to as information state or belief that compressed the data H_k and can be recursively computed using the last state and current observation as expressed in equation 3-23

$$b_{k+1} = \alpha p\left(Z_{k+1} \middle| x_{k+1}\right) \int_X p\left(x_{k+1} \middle| x_k, U_k\right) b_k \, dx_k \tag{3-23}$$

where α represent the normalization constant and the U_k can be generated based on the information state using a policy π_k as expressed in equation 3-24.

$$U_k = \pi_k(b_k) \tag{3-24}$$

the π_k represent the solution of a POMDP over a continuous observation space with a limitation that is intractable. In addressing this issue, Feedback based Information Road Map (FIRM) was proposed to minimize the intractable problem to a tractable POMDP by generating a representative graph in the information state space.

Given a FIRM graph with the controller, the policy π^g can be extracted by mapping graph nodes (v) to the edge (m) as expressed in 3-25

$$\pi^g : v \rightarrow m \tag{3-25}$$

Thus, the set of all graph planar (Π^g) generated in an information state-space allows POMDP to be tractable on the FIRM graph as expressed in equation 3-26

$$\pi^{g^*} = \arg\min_{\Pi_g} \mathrm{E} \sum_{n=o}^{\infty} C^g\left(B_n, \pi^g(B_n)\right) \tag{3-26}$$

where visited n-th node is represented by B_n., $C^g\left(B_n, \pi^g(B_n)\right)$ signifies the cost function. Experimental performance attained is impressive, but the limitation is the inability to cope with the dynamic environment. In their future work, they want to propose a framework that can learn and model changes using prior knowledge of object motion.

The technique proposed in the study of (Tian and Ma, 2016) to address the problem of SLAM is known as Double Guarantee Kidnapping Detection (DGKD). Comparing the DGKD with other SLAM techniques, 2 new processes were introduced to double-check and guarantee detection and their types. In carrying out this task, a threshold for the metric on real-time conditions was determined and introduced to avoid misjudgement. This increases the system reliability and performance. Figure 3-1 represents the DGKD workflow.

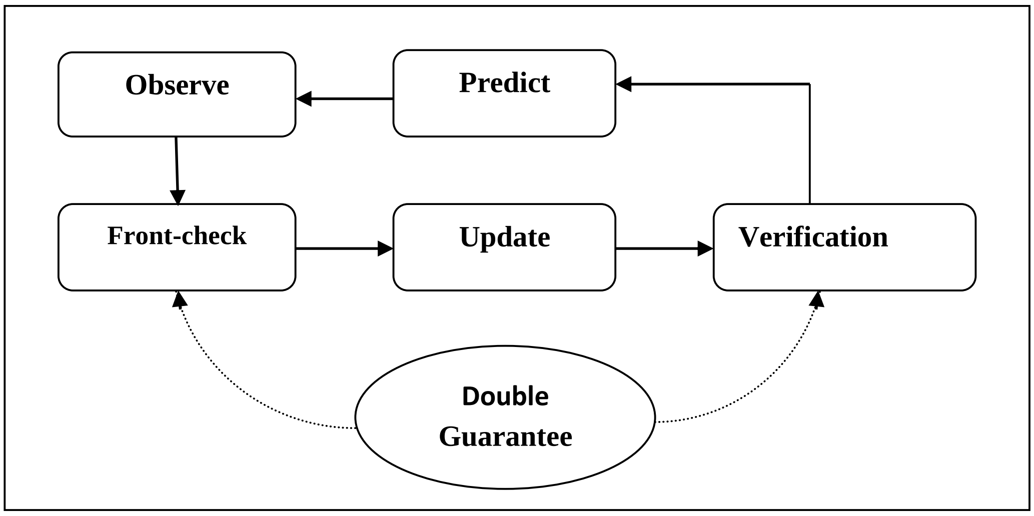

Figure 3-1: The overall workflow of the DGKD model (Tian and Ma, 2016)

However, DGKD limitation is the inability to cope with the relatively large-scale environment and to increase its adaptability in a large scale environment. The probability of features position and robot pose was combined to DGKD to form a new technique known as Probabilistic Double Guarantee Kidnapping Detection (PDGKD). In PDGKD, giving a robot state (X_r) expressed in equation 3-27 and the state of the feature (X_m) given in equation 3-28.

$$X_r = (x_r, y_r, \phi_r)^T \tag{3-27}$$

where (x_r, y_r) signifies position while ϕ_r denotes frame orientation of the robot,

$$X_m = (X_{m_1}^T, X_{m_2}^T, ...)^T, \tag{3-28}$$

while X_{mi} signifies the position of the feature (i) in the global coordinate as expressed in equation 3-29.

$$X_{mi} = \begin{bmatrix} \cos\phi_r & -\sin\phi_r \\ \sin\phi_r & -\cos\phi_r \end{bmatrix} \begin{bmatrix} {}^L x_i \\ {}^L y_i \end{bmatrix} + \begin{bmatrix} x_r \\ y_r \end{bmatrix} \qquad 3\text{-}29$$

${}^L x_i, {}^L y_i$ denotes the position of a feature i referred to the local coordinates frame attached to the robot.

Hence, the derived state vector combines both features state (X_m) and robot state (x_r) has expressed in equation 3-30 to address the DGKD problem of inability to cope with large scale environment.

$$X = \left(X_r^T, X_m^T\right)^T \qquad 3\text{-}30$$

In PDGKD, the operations are not only limited to the above-mentioned process, there are other procedures like prediction and updating operation similar to that of DGKD. The difference is the metric employed in their systems. Experimental comparison between the two techniques shows that PDGKD achieves a better result than DGKD. However, the limitation of the system arose when kidnap robot happens over a long period of time.

In the technique of (Tan et al., 2015), simultaneous localization and mapping without linearization were proposed to overcome the issue of linearization and inconsistency caused by approximation limiting the effectiveness of EKF-SLAM. This was achieved by combining Linear Time-Varying Kalman Filter (LTKF) and contraction tools on navigation problems with virtual measurements. Given a robot with LTV Kalman Filter SLAM using a virtual measurement in local coordinates, the EKF is the basis of LTVK and it would start with available non-linear measurement given in equation 3-31

$$\theta = \arctan\left(\frac{x_a}{x_b}\right) \text{ and/or } r = \sqrt{x_a^2 + x_b^2} \qquad 3\text{-}31$$

where θ represents the measure azimuth angle of the robot, (x_a, x_b) represent the location of landmark and r represents range measurement between robot and landmark. Afterward, the estimated Jacobian is employed to linearize this measurement to a locally stable observer. The expression of the relation in the Cartesian coordinate is given in equation 3-32.

$$h\mathrm{x} = \theta \quad \text{and/or} \quad h^*\mathrm{x} = r \qquad 3\text{-}32$$

But in 2D and 3D scenarios, the expression is given in equations 3-33 and 3-34

$$h = (\cos\theta, -\sin\theta) \quad \text{and/or} \quad h^* = (\sin\theta, \cos\theta) \qquad 3\text{-}33$$

$$h = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ -\sin\phi\sin\theta & -\sin\phi\cos\theta & \cos\phi \end{pmatrix} \quad \text{and /or} \quad h^* = (\cos\phi\sin\theta, \cos\phi\cos\theta, \sin\phi) \qquad 3\text{-}34$$

Given a linearize equation in local coordinate, the azimuth model representation for actual location of a landmark in 2D and 3D is expressed in equation 3-35 and 3-36

$$\mathrm{x} = (x_1, x_2)^T = (r\sin\theta, r\cos\theta)^T \qquad 3\text{-}35$$

$$\mathrm{x} = (x_1, x_2)^T = (r\cos\phi\sin\theta, r\cos\phi\cos\theta, r\sin\phi)^T \qquad 3\text{-}36$$

where h / h^* represents state independent measurement vector, ϕ represents the actual location of a landmark.

In LTVK, measurement and observation between θ and r are ignored, while feedback on tangential and Cartesian position errors between estimated and true landmark position (x) is taken into consideration. Therefore, the linear observation given above is substituted by virtual measurement, expression in Cartesian coordinate is given in equation 3-37

$$y = H\mathrm{x} + v(t) \qquad 3\text{-}37$$

where y represents observation/measurement vector which contains virtual/actual measurement, H represents the observation model matrix that includes state-independent measurement vectors and $v(t)$ represents zero-mean white noise. If further exploited, could be employed to attain a globally stable observer design with no set back error caused by linearization operation. However, this achievement is at a price of high computational cost and in their future work, they plan to reduce the computational workload by using the technique suggested by (Mu, 2013). The technique propose the use of landmarks with more provided information for feature selection to assist in reducing computational workload.

In the work of (Jia et al., 2016) vision-based technique using a monocular camera was proposed to initiate SLAM. The algorithm employed in their technique is known as Parallel, Tracking and Mapping (PTAM). In PTAM, procedures are split into two-level task operations in parallel. In the tracking level, the monocular camera fixed on a mobile robot is used to capture images from the environment. A ground feature-based pose estimation algorithm was proposed to detect ground features. Thus, achieving a more accurate robot pose relies on a weighted projection error-based energy function expressed in 3-38.

$$\min_{\mathrm{x}} \sum w(r_p) \| r_p(\mathrm{x}) \| \qquad 3\text{-}38$$

where w represents the Tukey bi-weight function for the homography-based projection error. Given the accurate robot localization attained by the expression in 3-38, matched features triangulated were used to generate an initial map and thereafter the second level operation known as mapping thread began. The mapping thread queries the initialized map and assists to incorporate new key features derived from using the epipolar searching procedure. The new matched features are selected for acceptance by searching for candidate region around the epipolar with minimal differences of Zero-Mean Summed Squared Differences (ZMSSD).

Candidate region with higher differences in ZMSSD compared to the threshold will be rejected for re-mapping. The expression is given in equation 3-39

$$s\left(u_j^i\right) = \begin{cases} r_j \leq \delta_i \ accept \\ r_j \rangle \ \delta_i \ reject \end{cases} \qquad 3\text{-}39$$

where

$$r_j = \pi\left(H_i v_j^i\right) - u_j^i ,$$

H_i represents the homography estimated by the RANSAC algorithm, v_j^i signifies the reference feature of u_j^i, r_j represents the projection error and δ_i represents the threshold.

After classification using equation 3-39, new map points generated are incooperated into the map points to improve the accuracy of the system. The indoor experimental performance carried out shows tremendous achievement towards accuracy. However, from the future work presented in their study, they intend to improve the performance of their technique to cope with mapping in various illumination scenarios.

The work presented in the study of (Agarwal and Burgard, 2015) is a graph-based simultaneous localization and mapping. The concept of graph-based SLAM relies on representing the nodes present in the graph by each pose attained by the robot. In real-world scenarios, these nodes can be used to signify features extracted from images captured by camera sensors or laser point clouds. Nodes can also be employed to signify physical landmarks of the object like trees, cars, etc. Edges present in the graph are signified by a factor connecting two nodes. These factors represent the bearing measurement of features. Given a robot navigating in an unknown environment using graph-based SLAM, the first problem to address is creating a graph. This can be attained by identifying the nodes and the factors connecting them on the data generated from the sensors. This computation is known as front-end. The second problem to address is the node's configuration that provides the best explanation for the factors. These steps assist to compute a maximum likelihood map and this computation is known as back end. Thus, the back end aim is to find the configuration of nodes that minimize error created by the factors from the front-end operation. If $x = (x_1,........,x_n)^T$ represents a state vector and x_i signifies the pose of a node i that can also represent a robot or landmark position. The error function $(e_{ij}(x))$ description for a single factor between i and j represent the difference between Z_{ij} and $\hat{Z}(x_i, x_j)$. An expression is given in equation 3-40.

$$e_{ij}(x) = \hat{Z}(x_i, x_j) - Z_{ij} \qquad 3\text{-}40$$

where $\hat{Z}(x_i, x_j)$ signifies the expected measurement given in the current state, Z_{ij} represents the observed measurement, i and j represents the graph nodes. The re-projection error of the observed landmark must be minimized for accuracy purposes. Thus, the minimization error can be expressed in equation 3-41

$$\mathrm{x}^* = \arg\min_x \sum_{ij} e_{ij}(x)^T \textstyle\sum_{ij}^{-1} e_{ij}(x) \qquad 3\text{-}41$$

where $\sum_{ij}^{-1}$ represent information matrix related to the factors that exist between two poses x_i and x_j, $\sum_{ij}$ signifies the covariance matrix and x^* represent the optimal configuration of nodes with limited error induced from the factors of front end operation.

However, the effectiveness of the graph-based technique towards SLAM has attracted researchers like (Agarwal and Burgard, 2015). In their enhanced graph-based SLAM, the experimental result shows tremendous success towards SLAM problem. Thus, the computational cost is high due to an increase in computational requirement at the matrix factorization stage. Furthermore, their graph-based SLAM couldn’t cope with the dynamic environment and in their future work, they will be improving the SLAM technique towards tracking the dynamic environment.

The technique proposed in the work of (Li-Chee-Ming and Armenakis, 2016) is to address the problem of SLAM based on the combination of Visual Servoing Platform (ViSP) and Red, Green, Blue and Depth-SLAM (RGBD) techniques. The ViSP is a commonly used open-source tool for tracking relative pose between the camera and the model of an object (Martinet et al., 1997). The VISP is capable of carrying out the task of extracting the features important to address the problem of SLAM. It entails the Moving Edge (ME) algorithm proposed by (Bouthemy, 1989) for feature extraction by matching the feature point in an image and projected model. Hence, the current pose of the camera is estimated using a non-linear optimization method known as virtual visual servony. However, relying on ViSP alone might not be sufficient because the issue arose when tracking is missing and there will be a need for re-initialization. This issue can occur as a result of a lack of model features in the sequence of the image captured by the camera, rapid camera motion can also be a contributor to loss of tracking. Therefore, improving the tracking becomes important and RGB-D SLAM was proposed concurrently to provide the ViSP with the missing tracking. The RGB-D SLAM is a graph-based approach similar to the one proposed in the work of (Agarwal and Burgard, 2015) which involves the frontend and backend operation. Figure 3-2 is a model representing the integration workflow between ViSP and RGB-D SLAM.

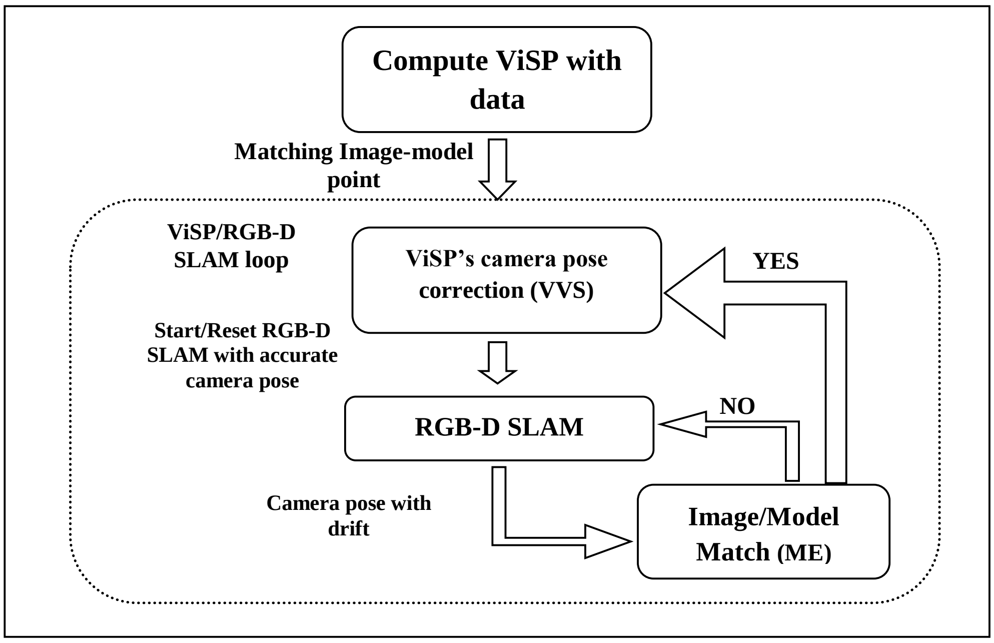
Compute ViSP with data
Matching Image-model point
ViSP/RGB-D SLAM loop
ViSP's camera pose correction (VVS)
YES
Start/Reset RGB-D SLAM with accurate camera pose
NO
RGB-D SLAM
Camera pose with drift
Image/Model Match (ME)

Figure 3-2: RGB-D SLAM /ViSP integration workflow (Li-Chee-Ming and Armenakis, 2016)

The collaboration was successful and experiment performance carried out on the second floor of York University, Bergeron Centre for Engineering and Excellence shows tremendous improvement towards the recovery of lost tracking but at the expense of high computational cost. The experiment further revealed that the computational issue did not happen at the ViSP operation. Rather it happened at the RGB-D SLAM running without resetting for an extended period of time, due to processing simultaneously,a huge size of data. However, in the work of (Agarwal and Burgard, 2015), they also proposed the RGB-D graph-based SLAM technique and experience the issue of computational cost. In conclusion, It could be that computational issue is a general problem associated with RGB-D SLAM technique.

In the work of (Irie et al., 2012.) they proposed to address the problem of SLAM for outdoor navigation taking into consideration drastic illumination changes which happen in most environments. In this technique, the stereo camera capable of obtaining 3-D (3 dimensions) ranges data was employed to capture data from the environment. Afterward, a 2-D (2 dimensions) grid map that is not much affected by illumination condition is generated. Given the 2-D grid map, occupancy information and salient line segment can be extracted perfectly. The particle filter is employed to extract the robot pose while edge point based stereo SLAM was used to obtain robot ego-motion and the occupancy information simultaneously. This extracted information is used to address the SLAM problem. There are other important procedures carried out to develop their technique. The model in Figure 3-3 provides a full description of the proposed SLAM technique for mobile robot navigation in an outdoor environment.

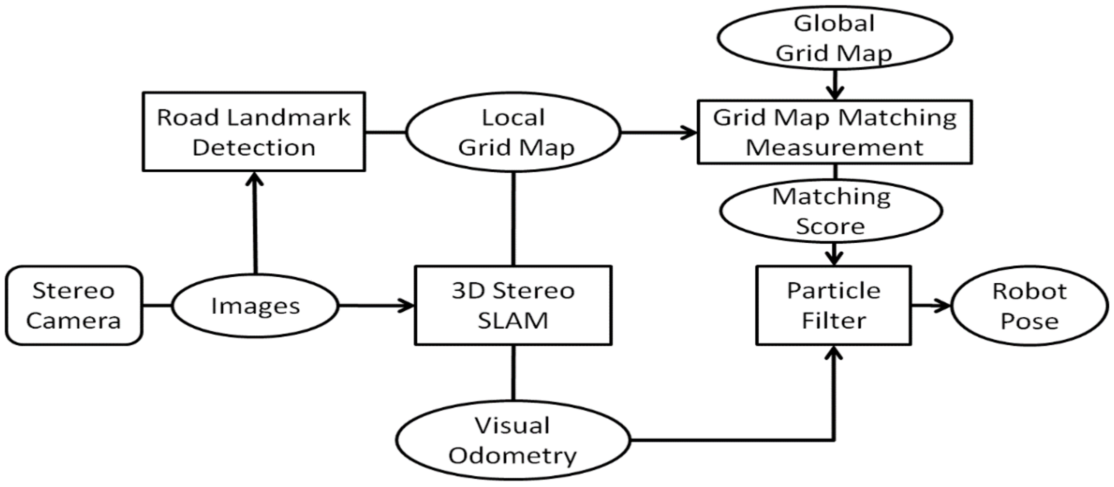

Figure 3-3: Proposed stereo SLAM technique for a mobile robot in an outdoor environment (Irie et al., 2012.)

The proposed model was successfully implemented and the experimental performance shows their visual odometry recovering from error and performed well under various illumination situations. However, the technique failed under extremely adverse illumination conditions such as when direct sunlight covers a large part of the image. Giving such condition, limited edge point is extracted for detection, which resulted in a huge error in motion estimation and inability to recover from kidnap robot.

In the research work of (Oh et al., 2015), they proposed to develop a Simultaneous Localization and Mapping (SLAM) technique, using a monocular camera and a 2-D laser scanner sensor. The two sensors were encouraged because of an environment with ambiguity such as long corridor, SLAM algorithm working with laser scanners might not be able to estimate correctly the robot position. In resolving this problem, a monocular camera was introduced into their system to collect data from the environment. Figure 3-4 illustrates how the proposed SLAM technique makes use of the two sensors

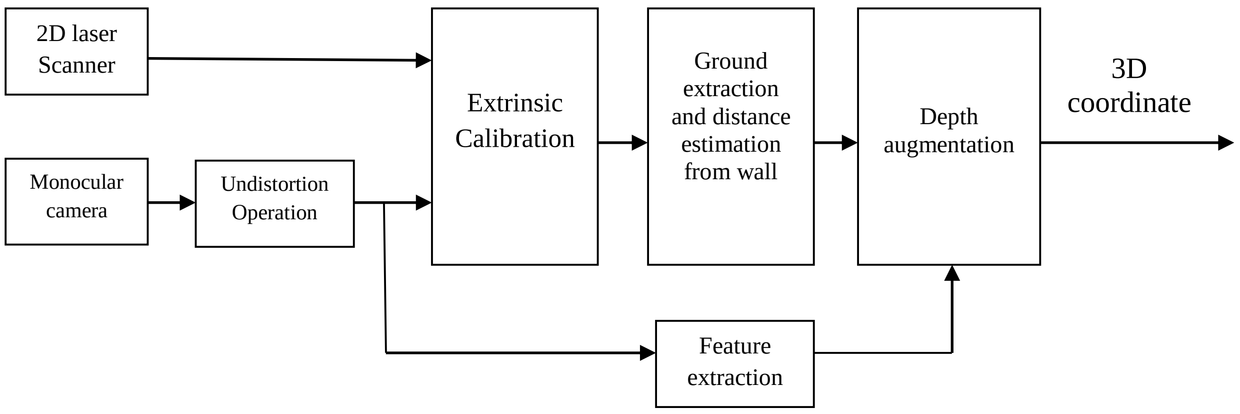

Figure 3-4: The proposed monocular camera and laser scanners sensors based SLAM technique (Oh et al., 2015)

At the monocular camera stage, graph structure-based technique and hybrid method which allows the SLAM technique to estimate the robot pose in ambiguity environment was proposed to handle where the laser scanner fails. In a graph-based technique, the SLAM problem using a conditional probability is expressed in 3-42.

$$p(x|z)\alpha \prod_{i} p(z_i|x) \qquad \text{3-42}$$

where x signifies the robot pose, $z = (z_1,........z_n)$, z_i signifies measurement of a sensor at i^{th} step while $p(z_i|x)$ represents a potential function.

Given this technique, an experimental comparison was carried out with a conventional G-mapping approach and results revealed that their system performs better than the G-mapping approach. However, as mentioned in the section of their results, the algorithm proposed has a small computational burden because more image processing is executed when a node is added.

This was also supported to be true because, in the work of (Deming and Perlovsky, 2006). It was stated that data association from multiple sensors can cause the computational complexity of systems to be prohibitively high with an effect that can lead to system failure.

The study of (Clipp et al., 2009) proposed a Simultaneous Localization and Mapping (SLAM) technique using a stereo camera as the source of retrieving information from the environment. In their technique, they took into consideration the issue of computational cost. They propose the use of Kanade-Lucas-Tomasi (KLT) feature tracking because of its advantage of high-speed in processing and robustness to features that are repetitive in nature. This technique was combined with a wide baseline feature which further helps the system to improve its robustness to repetitive features and allows it to recognize previously visited areas in the environment. The experimental performance of the system is impressive. However, as stated in their conclusion, an item on the desk in front of the windows appeared blank due to the presence of high intensity of bright sunlight. Furthermore, the system was unable to recognize the area that has already been mapped due to movement of objects in the same area. This is as a result of low dynamic range of the system. Thus, these two problems when encountered, make their system fail. In the future, they plan to address this issue by extracting features in a 3D scene just as it was proposed in the work of (Changchang et al., 2008). They tried to combine 3D geometry and sparse feature detection with the aim of using immovable features such as floor, wall and ceiling to re-localize itself regardless of dynamic changes happening in the environment. Successful implementation of this idea will allow the system to overcome the issue of a dynamic environment and illumination variances

In the work of (Lin et al., 2013), they proposed a robust outdoor simultaneous localization and mapping (SLAM) by using a stereo camera for collection of data and EKF is employed for tracking and updating of a feature. The SLAM technique's main target is to cope with directional sunlight illumination causing shadows and irregularity in various lighting scene.

Two novel methods are introduced to improve the robustness of their SLAM technique. Locally maximal features selection technique based on Harris corners was employed to extract edges in inconsistence illumination scene and shadows. The second method is the 3D feature/landmark matching, which assists to improve the robustness of the feature/landmark matching. Figure 3-5 shows the overview flow of the proposed SLAM algorithm.

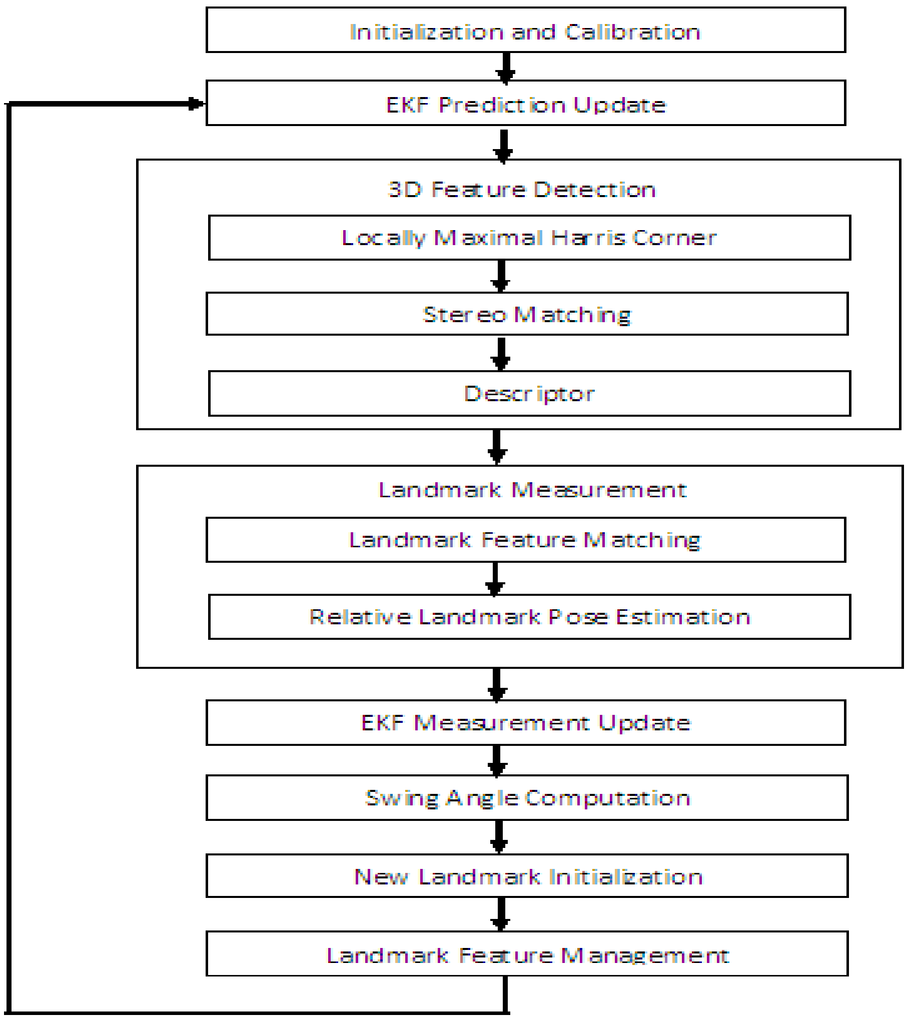
Initialization and Calibration
EKF Prediction Update
3D Feature Detection
Locally Maximal Harris Corner
Stereo Matching
Descriptor
Landmark Measurement
Landmark Feature Matching
Relative Landmark Pose Estimation
EKF Measurement Update
Swing Angle Computation
New Landmark Initialization
Landmark Feature Management

Figure 3-5: Flow of operation for the Proposed SLAM Technique (Lin et al., 2013)

Given the proposed SLAM technique, experimental comparison supports the SLAM technique ability to select features evenly across the image and more consistent in various illumination scenes and shadow compared to the commonly used Threshold method. However, the proposed

SLAM technique suffers from high computational cost and they desire to reduce it by minimizing the search region using a feature match search window technique. They also want to introduce multi-threading algorithms to allow multiprocessing to be carried out, this procedure will also improve the processing speed of the SLAM technique.

In the study of (Gao and Zhang, 2015) an RGB-D SLAM system was proposed to overcome SLAM Problem. An RGB-D camera sensor is a common sensor in SLAM research, because of the ability to provide vision and depth information about features. This information can be processed by Iterative Closet Point (ICP) algorithm to align features position where the spatial point is extracted using the correlating depth data of the sensors. Meanwhile, the depth measurement for feature position is often affected by noise, instead, they propose the use of planar features to extract reliable depth values. Given any RGB-D SLAM system, it consists of two Parts: the graphic end sometimes called front end, which performs the operation of image acquisition, processing Image, object/features extraction, loop closure detection and frame alignment while the second part is known as optimization end / back end which performs the operation of integrating new observation into global model. Figure 3-6 shows the schematic overview of the RGB-D SLAM system.

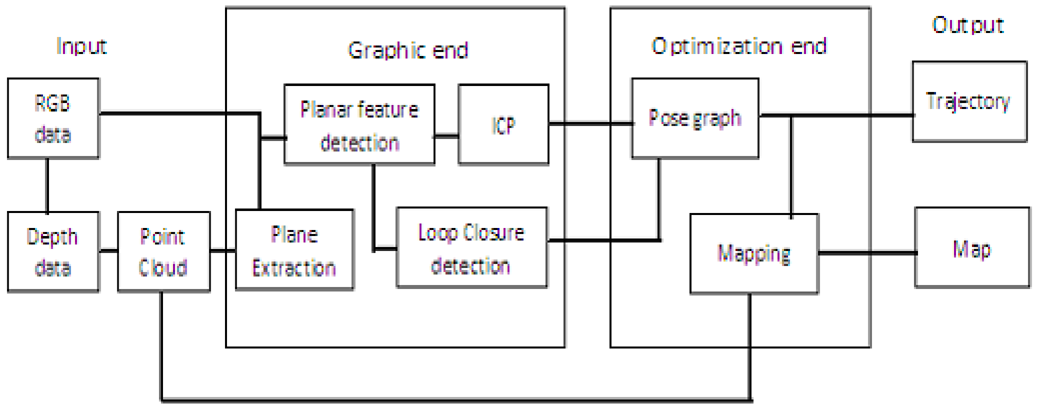

Figure 3-6: Schematic overview of the RGB-D SLAM technique (Gao and Zhang, 2015)

The experimental performance of the proposed RGB-D SLAM technique produces a tremendous result, but there are many instances where the system fails to attain good results. The long corridor with a repetitive structure on the floor and wall contributed to false positive detection. Besides these factors, reflection from the ceiling, disturb the matching process while direct sunlight from the windows also contributed to the SLAM system failure.

The study of (Yi and Wang, 2015) propose an autonomous robot path planning and localization technique based on potential field for generating a quality map in a static environment. The proposed SLAM system state relies on the robot position and the observed coordinate of features. These data are analyzed by the potential field algorithm, six procedures are implemented to achieve a detailed map for the static environment. The determining of robot control law, the state prediction, the environment observation, the data association, the state vector/covariance matrix update and the map building phase. Figure 3-7 shows the procedural flow of the proposed potential field SLAM

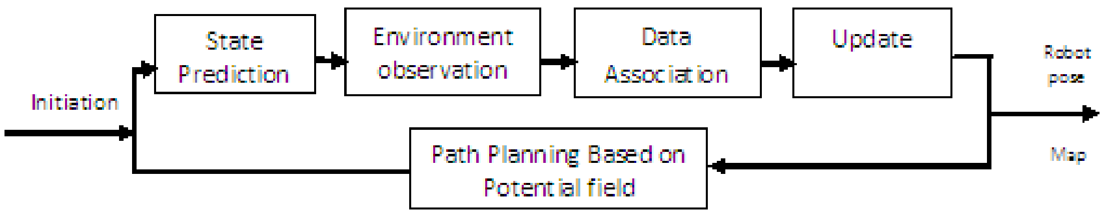

Figure 3-7: Operational flow for potential field SLAM (Yi and Wang, 2015)

The proposed SLAM technique was tested in an environment where features are distributed evenly and 7 waypoints are employed to guide the robot trajectory. The result obtained from the test is remarkable and this supports its capability towards addressing the SLAM problem. However, the proposed algorithm is only capable of functioning perfectly in a static environment. The future work is directed towards improving the algorithm's ability to cope in a dynamic environment.

The study of (Bowman et al., 2017) proposes a different SLAM technique compared to the traditional technique of SLAM that depends on geometric features like edges, lines, plane and points. Instead, they employed a semantic SLAM technique. Given that the state ℓ of each landmark consists of its position $\ell^P \in R^3$ as well as a class label ℓ^C from a discrete set $C = \{1, \ldots.., c\}$. The SLAM problem is addressed by estimating the landmark state and sensor trajectory using 3 sources of information such as semantic object observation, inertial information and geometric point features. The semantic SLAM was also enhanced by the use of the expectation-maximization algorithm to effectively handle the semantic data association. This algorithm was implemented in C ++ using Georgia Tech Smoothing and Mapping (GTSAM) and Incremental Smoothing and Mapping (ISAM) (Dellaert, 2012) for the implementation of the back end optimization. The dataset used for testing is collected from Kitti outdoor dataset using odometry sequence 05 and 06. This dataset was selected because it is common and will assist in comparison with another researcher SLAM technique. Experimental comparison of the proposed technique as compared to Oriented fast and Rotated Brief (ORB) SLAM. The results obtained shows better performance because the lack of inertial information in ORB SLAM leads to frequent lost and many missing trajectory estimates. Observations support the proposed technique ability to handle repetitive structure in the environment than ORB SLAM. However, in their future work, they plan to extend the ability of their algorithm to estimate the full pose of the semantic object. They also plan to introduce multiple sensors, and further address the issue of the non-stationary objects in the environment.

The work of (Saleem, 2013) proposes an effective technique towards addressing the problem of SLAM, using a mobile robot equipped with a single Ultrasonic Range Finder (URF). The data generated from the URF consist of the radial distance and corresponding angles to object, while the data generated from the digital compass reveal the pose of the autonomous robot. The data generated from the optical assembly communicated precisely the displacement of the autonomous robot. Thus, the AI algorithm is used to analyse the data generated to create a finalized map of the environment. Figure 3-8 illustrate the flow chart of the proposed sonar SLAM technique

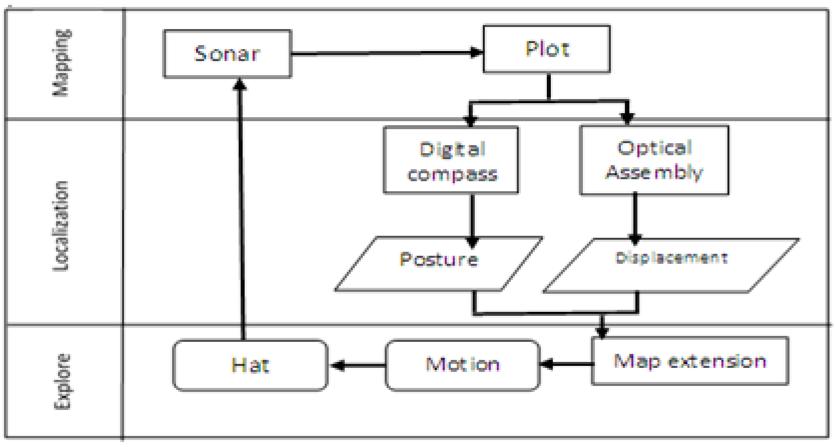

Figure 3-8: The flow chart of sonar SLAM technique (Saleem, 2013)

The proposed sonar SLAM technique was tested on three different occasion but the sonar SLAM attained an outstanding result. However, in the attempt to improve its accuracy, when increasing the resolution by decreasing stepper angle, the computational cost is high, and this requires a larger amount of time to generate SLAM data for processing.

The study of (Zhang et al., 2017), proposed an advance visual-inertial SLAM system with flexible sensor fusion and hardware known as Percept in Robotics Vision System (PIRVS). The hardware in PIRVS is equipped with 2 Omni-direction Complementary Metal Oxide Semiconductor (CMOS) image stereo Sensors capable of capturing colour image at a resolution of 640*480. An inertial measurement unit and a multicore processor were used for processing image that was captured. Thus, three major components are also involved in PIRVS algorithm. An image processing front-end used for extracting image features and matching. EKF-based visual-inertial odometry SLAM algorithm for feature tracking, creating and updating map with prior poses from the tracking thread. Figure 3-9 shows the PIRVS system architecture.

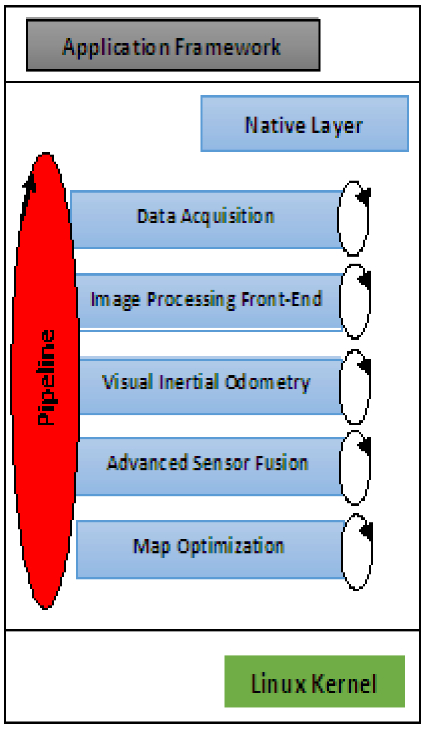

Figure 3-9: Overview of the PIRVS system architecture (Zhang et al., 2017)

The proposed PIRVS system was evaluated and compared with the system of Key-frame Visual Inertial System (KVIS), Visual Inertial System (VINS) MONO using a dataset from PENN COSIVIO (Zhang et al., 2017) based on accuracy and processing speed. In both testings, the PIRVS system attained a better result than KVIS and VINUS-MONO. However, the proposed PIRVS system perform woefully in low lighting condition compared to KVIS and VINUS-MONO. In future work, they plan to improve their SLAM algorithm by introducing sliding windows of poses to enhance their image processing phase, to cope, when facing challenging illumination scenes.

The study of (Al-Mutib, 2015) proposed a SLAM technique for autonomous navigation for a mobile robot using an active stereo sensor for vision. Monte Carlo localization is used for analyzing the stereo images. Its operation is in two phases: The prediction phase where the set of selected particles computed from the previous iteration are applied to the motion model and updating phase where the observation measurement and weight of samples are taken into consideration. The resampling at this phase selects particle with a higher probability of likelihood associated with them, to generate a new set of samples. The proposed vision technique is not only limited to SLAM but obstacle detection and avoidance are also incorporated into their system. Figure 3-10 shows the framework for stereo vision used for multiple purposes.

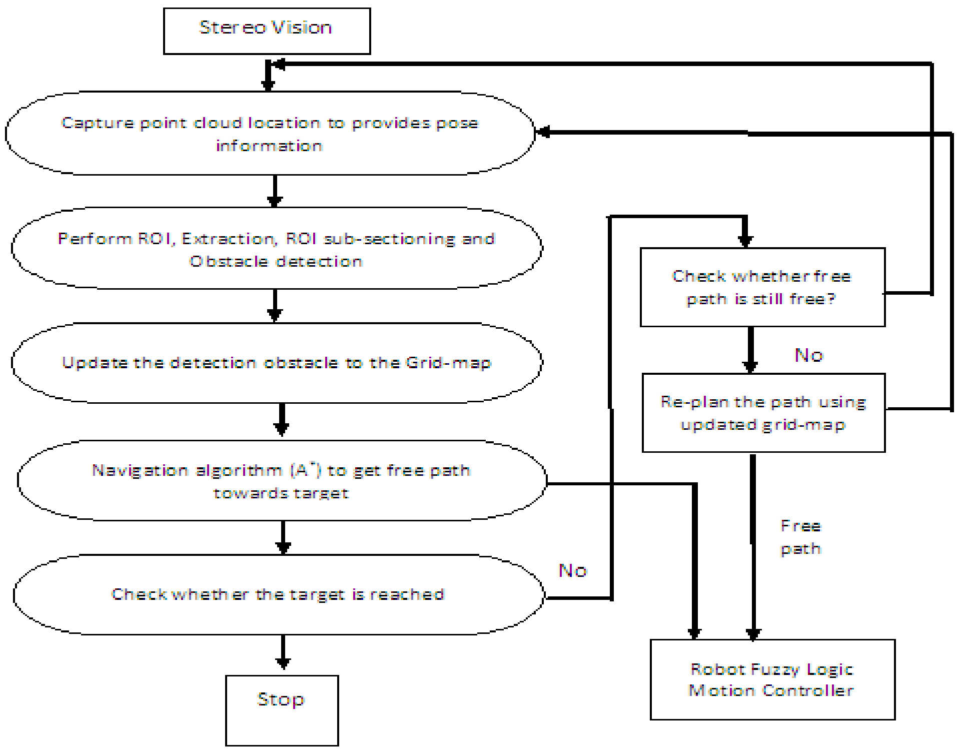
Stereo Vision
Capture point cloud location to provides pose information
Perform ROI, Extraction, ROI sub-sectioning and Obstacle detection
Update the detection obstacle to the Grid-map
Navigation algorithm (A*) to get free path towards target
Check whether the target is reached
Stop
No
Check whether free path is still free?
No
Re-plan the path using updated grid-map
Free path
Robot Fuzzy Logic Motion Controller

Figure 3-10: Framework for stereo vision multi-purpose (Al-Mutib, 2015)

The proposed technique was tested and the experimental result of the robot pose error compared to ground-truth classification is outstanding. It also proves its ability to manoeuvre in real-time, given an unstructured and complex environment present with dynamic objects. However, data association is challenging when image scene possesses the characteristic of illumination variation, specular reflection and the inconsistency in point cloud because of the variation in viewing angles of the stereo camera sensors. They plan to address the lighting variation in their future work so as to improve the performance of the technique at the data association phase.

The study of (Tan et al., 2013) proposes a monocular SLAM technique that can robustly function in a dynamic environment. Their SLAM technique is different from the traditional Parallel Tracking and Mapping (PTAM) method because it identifies the dynamic regions from the static regions. This was achieved by an online framework capable of updating and representing a model for a dynamic environment where a change in appearance is easily detected and properly handled. Figure 3-11 represents the framework used for their study. In addition, they introduced a new prior based adaptive Random Sample Consensus (RANSAC) algorithm known as Prior-based Adaptive RANSAC (PARSAC) into their SLAM system. This assist the system in handling fast camera movement even in challenging situations like illumination variation in the image scene.

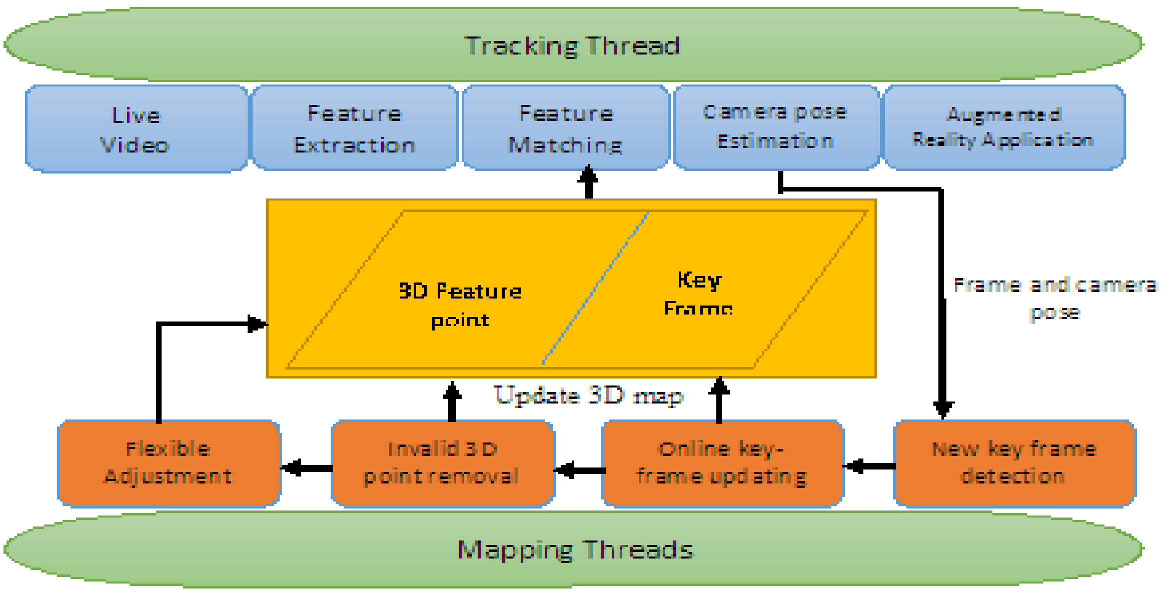

Figure 3-11: Overview framework of PARSAC algorithm (Tan et al., 2013)

Given the proposed PARSAC, experimental comparism was carried out with the RANSAC algorithm and the result obtained shows that PARSAC algorithm has better performance than the RANSAC. Furthermore, it was also compared with PTAM and the result supports that PARSAC algorithm has the capability to cope with fast camera movement, global re-localization and proper handling of the dynamic environment than PTAM. However, the proposed PARSAC SLAM technique has some limitations, for instance, when an object is moving too fast eventhough it meant to cope with the dynamic environment, the SLAM technique is likely to produce a wrong measurement. Moreover, the computational cost as related to real-time over the large-scale scenes is not efficient and in future work, their target is to improve the system ability to work efficiently in larger space.

The research study of (Gomez-Ojeda et al., 2017) proposed a stereo camera SLAM technique that relies on point features from camera for trajectory estimation and map building of an environment. This technique is known as Point and Line SLAM (PL-SLAM). In PL-SLAM, the stereo vision system combines the line segment and points features to function effectively over a wide variety of image scenes. This technique was proposed so that each method can cover up for each other's failure particularly in image scenes where point features are not properly distributed or scarce, the line segment is employed for carrying out SLAM activities and vice versa. The PL-SLAM employs ORB algorithms for keypoint detections and key point matching, while the line segment detection algorithm was employed to extract the line segments. Given the point feature and line segment, the Kalman Filter was employed as the SLAM algorithm to analyse these factors to generate a detailed map for the environment. Figure 3-12 shows the flow of operation for the PL-SLAM technique.

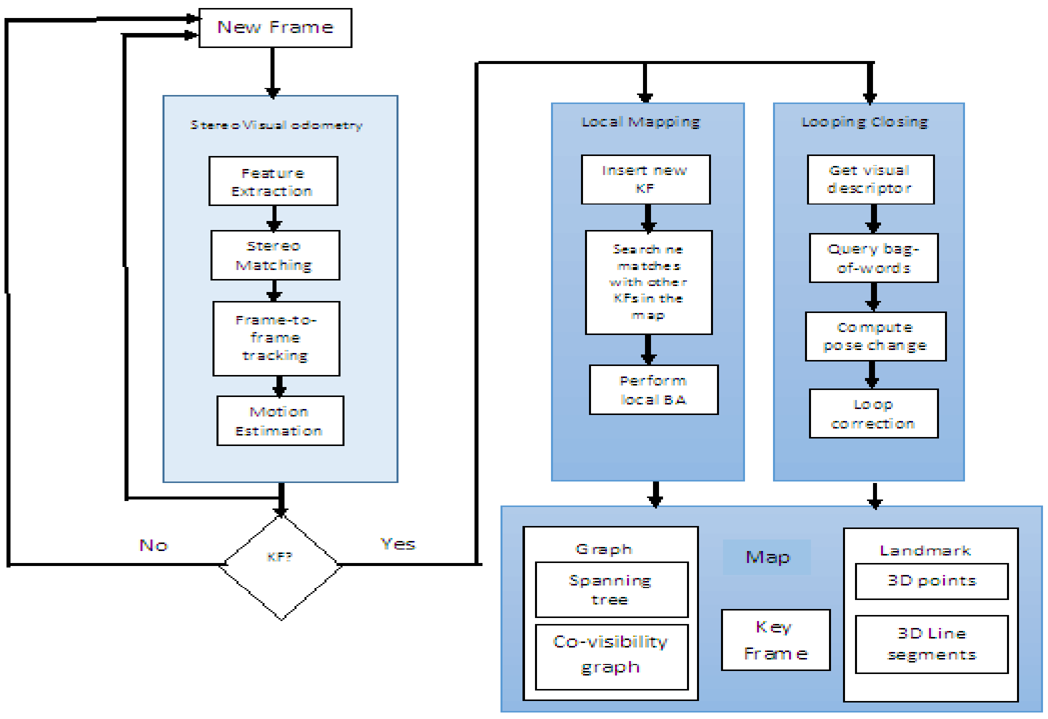

Figure 3-12: The operational flow of PL-SLAM technique (Gomez-Ojeda et al., 2017)

The proposed PL-SLAM was tested with various data set such as KITTI and EUROC. It was developed with C++. An experimental comparison was carried out with other SLAM techniques such as L-SLAM, P-SLAM and ORB-SLAM. The result obtained from most categories of testing shows that PL-SLAM outperforms other techniques. However, in their future work, they aim to adopt the technique proposed in SVO and PL-SVO to reduce the computational time at the feature tracking phase by estimating the position of the feature from motion estimation.

The research work of (Engel et al., 2015) proposes a Large Scale Direct SLAM technique based on stereo vision (LSD-SLAM) that relies on Kalman Filter for feature tracking and local mapping. This proposed technique has the ability to run in real-time on the standard central processing unit. The LSD-SLAM function characteristics are contrary to Sparse Interest Point SLAM (SIP-SLAM). The LSD-SLAM technique aligns frames of images using the photo consistency of high texture areas, as well as edges, corner and high contrast pixels. It simultaneously calculates the pixel depth using two stereo cues: static stereo which is obtained through the fixed baseline stereo sensor setup and temporal multi-view stereo obtained during camera motion. The fixed baseline was employed to avoid scale drift error that commonly happened in pure monocular SLAM technique and the affine lighting correction technique was proposed to handle aggressive illumination variation changes between frames of images. Figure 3-13 illustrates the overview of the stereo large scale direct SLAM technique.

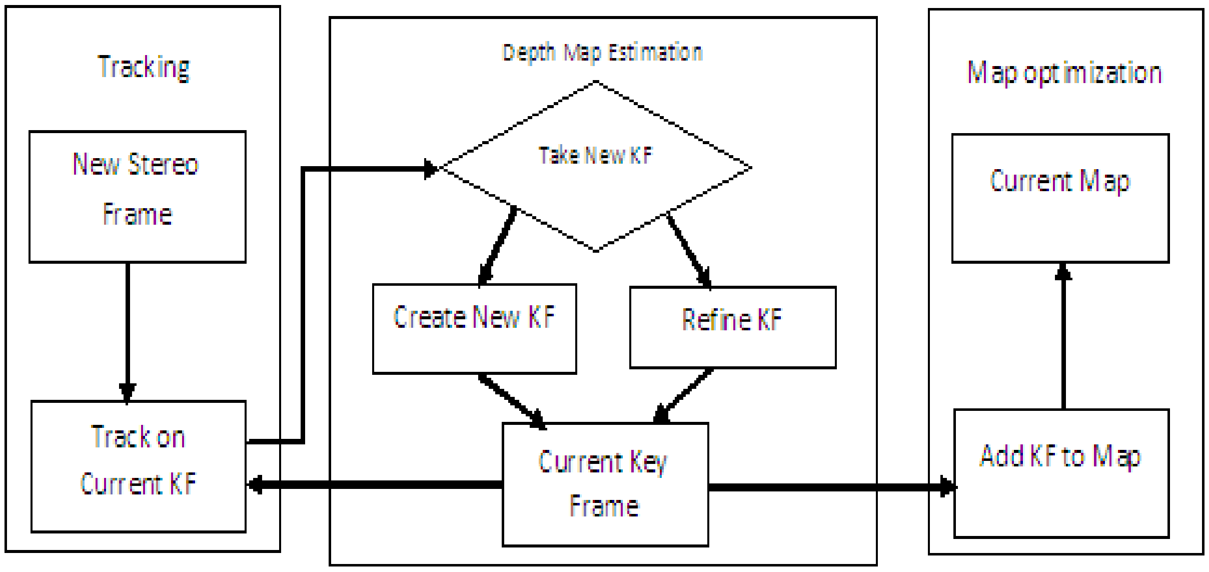

Figure 3-13: The overview of LSD-SLAM technique (Engel et al., 2015)

Given the proposed LSD-SLAM technique, the experimental performance was carried out using the popular KITTI benchmark dataset for stereo odometry and SLAM on autonomous cars. Experimental results are discussed qualitatively and quantitatively. However, in both experiments, the outcome obtained is impressive, but the technique failed in some certain conditions that they intend to solve in their future work. They plan to improve the robustness of their technique by taking into consideration the multi-body motion segment and estimation, which will assist the system to cope with the independent dynamic object and dominant motion in images.

In the work of (Khan et al., 2018), the autonomous robot developed is required to navigate in an unfamiliar location by building a map of its environment by means of the data collected from the sensors and using the map to find its location at minimum error. The sonar sensor was employed and the occupancy grid mapping technique was used for the analysis of data to generate the map. In addition, the relative positioning approaches that assist with a location using a wheel encoder and inertial measurement unit based on the Kalman Filter was also introduced into the SLAM system. The discrete step-wise modelling was also used to guide its navigation. All proposed techniques are implemented on a Raspberry Pi, which represents the main computational block of the robot. The Raspberry Pi 3 model has a 1.2 GHz 64-bit quad-core Advanced RISC Machine (ARM) V8 CPU with 1GB RAM. The proposed SLAM system was able to cope with various issues like the dynamic environment, actuator faults, computational cost and error in the motion model. However, the system is only limited to the indoor environment and an attempt to test it in an outdoor environment makes the system fail. Another limitation is the use of ultrasonic range sensors, which are expensive compared to other sensors like Lidar and camera. Lastly, the issue of kidnap robot where the robot lost position due to the accumulation of error also limits the performance of the SLAM technique proposed. In the future, all limitations are planned to be addressed in their new SLAM technique.

The study of (Wang et al., 2014) proposed a localization technique based on particle filtering for an autonomous robot in high occluded conditions and dynamic environment taking moving people into consideration. In this technique, Proposal Distribution Function (PDF) of the particle is estimated using odometer but given the condition of high occlusion and intense dynamic environment. The PDF-based odometer might not provide accurate particle distribution. This limitation is minimized by proposing the Localization based Particle Filter with Roulette re-sampling (LPF-r) to maintain accurate robot pose in such environments. The

LPF-r is a framework based on Particle Filtering with Roulette re-sampling (PF-r) and the architectural design is given in Figure 3-14. The concept is to provide a strong localizability, which will reduce the influence of dynamic environment and occlusion on localization

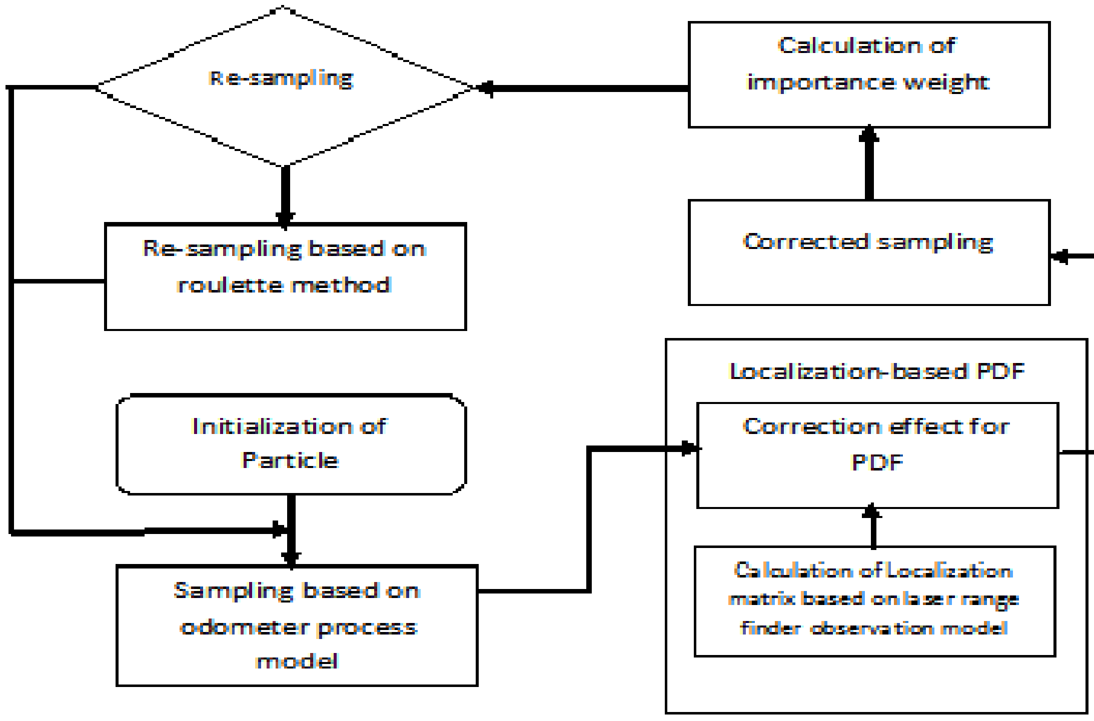
Re-sampling
Calculation of importance weight
Re-sampling based on roulette method
Corrected sampling
Localization-based PDF
Correction effect for PDF
Initialization of Particle
Calculation of Localization matrix based on laser range finder observation model
Sampling based on odometer process model

Figure 3-14: Represent the architecture framework of the localization-based particle filter with roulette re-sampling (Wang et al., 2014)

Given the proposed technique, experimental results show the necessity of considering the influence of prior-map during localization. It also supports the proposed algorithm capability to maintain accurate robot pose, even in the presence of high occluded and dynamic environment at an acceptable real-time. However, the kidnap problem contributes to system failure and occurs when the odometer error and occlusion are caused by a dynamic obstacle.

Considering the observation from the review conducted, this study suggested that new researchers must focus on the high computational cost as related to processing time. This is a major problem mostly complained by researchers reviewed in this study. In addition, other problems such as illumination variance (light intensity and shadow), kidnap robot and dynamic environment are persistent as mentioned in this review by many researchers. Figure 3-15 shows the overall impact of these SLAM problems as related to this research.

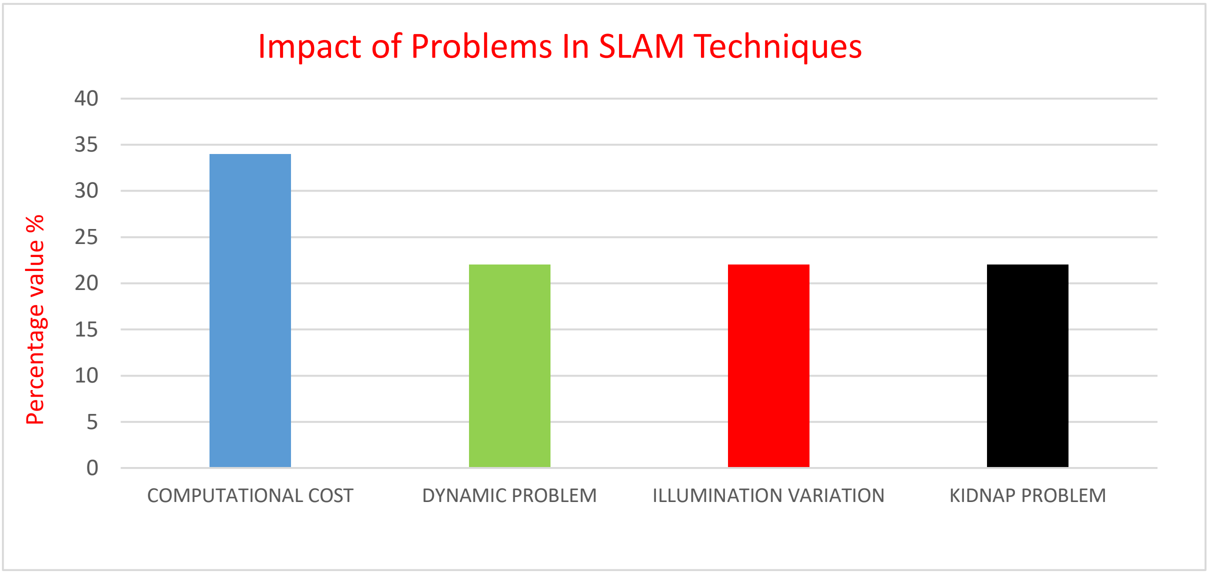

Figure 3-15: The overall impact of the SLAM problems

In Figure 3-15, the computational cost represented in blue chat has the highest percentage value of 34, implying that 34% of the reviewed papers complained about the computational cost, while dynamic problems, illumination variation and kidnap problem have 22% value each, implying that 22% of all reviewed papers complained about the dynamic, illumination variation and kidnap problems respectively. Therefore, the above-mentioned problems were considered and a DIK-SLAM technique with multiple re-modification of the algorithm was presented to eliminate each of these problems. However, considering the re-modification, the computational cost of the proposed SLAM is taken into consideration to avoid slow processing speed even though it will enhance the performance of the algorithm. Therefore, the system must be monitored for accurate trajectory in real-time, that will facilitate perfect navigation for autonomous robots. Successful accomplishment of the study would tremendously contribute to SLAM research because this would facilitate robot self-exploratory expeditions, which will reduce human risk in an unstable environment.

3.3 Challenges, Open issue and Research Direction

The SLAM domain has experienced a lot of research and various effective methods have been recommended. Nonetheless, there is no general technique of handling the complex problems various researchers experience during the implementation of their SLAM techniques, but there has been improvement towards addressing some of these issues. The example could be seen on the issue of illumination variance, which happens when an object is obstructing light generated from a source, this problem has not been fully addressed. However, to some extent in static environment, the effect has been minimized in the work of (Agunbiade et al., 2014, Makhubela et al., 2018). In a dynamic environment, the effect of illumination variation becomes difficult to address. This happens in the condition when a static objects cast dynamic illumination, the localization becomes difficult to estimate . It is even more problematic and challenging to estimate localization if a dynamic object is casting a dynamic illumination (Le Cras et al., 2013). A different problem that has not been fully resolved is the kidnap problems, although it has received a lot of attention from researchers with improvement thereon. The kidnapping problem occurs when the robot moves from one position to another without having any information about its new position (Guyonneau et al., 2012) and this can happen during failing sensors or an increase in measurement noise (Negenborn et al., 2003). Meanwhile, when a robot is kidnapped in an environment, algorithms selected to solve this problem must carry out these objectives. The ability to perform pose estimation, detect kidnap problem and global localization (Guyonneau et al., 2012). However, in the literature, the initializing localization technique has been positive to kidnap robot (Se et al., 2001). In this technique, an extensive search of the current observation over the reference map (pre-mapped environment) will be carried out to find the robot position in relation to this map. This will assist the robot to start up again at the last stop position since the last position has already been mapped in the previous map. Thus, in a dynamic condition that the current observation has changed from the previously referenced map, it becomes challenging to address because it will not locate its current observation from the referenced map, and re-localization becomes impossible, which might lead to robot lost without recovery (Se et al., 2001). However, the dynamic environment which contributes towards complicating other problems is also an open problem that researchers are still studying to date. Achievement in this research area depends on how dynamic the environment is, the more dynamic the environment, the more difficult and challenging it is to address Lastly, conducting a review in SLAM is incomplete without mentioning the computational cost as related to real-time. This is widely complained by previous researchers working on SLAM (Agarwal and Burgard, 2015, Castellanos et al., 2001, Oh et al., 2015). In literature, some researchers have proposed a wide range of algorithms with mathematic technique to attain impressive result towards computational cost (Dellaert et al., 2010, Kaess et al., 2008, Konolige et al., 2010). These algorithms have their own advantages and disadvantages (Dissanayake et al., 2011). Take for instance, the Fast-SLAM commonly used by researchers (Montemerlo and Thrun, 2003, Qiu et al., 2012) trying to limit computational cost which relies heavily on particles. The accuracy of the Fast-SLAM is dependent on the number of particles which can also increase the computational complexity (Wurm et al., 2003). Some researchers proposed to improve the accuracy by introducing more algorithms to enhance the SLAM technique performance, but it increases the computational complexity (Li-Chee-Ming and Armenakis, 2016, Yinka et al., 2014). In both situations, it becomes a trade-off between accuracy and SLAM runtime. Therefore, researchers are to decide whether to reduce the accuracy of the SLAM system to decrease its computational cost or vice versa. In future, the greatest accomplishment would be to attain a global optimal solution that would address all SLAM problems. In attaining this goal, more investigation must be carried out for proper understanding of all problems. Given a robot in real life scenario, it should be able to localize and create map for environments like in-door, out-door and under water. In any environment, objects are often present, some are static while others are dynamic in nature. Therefore, SLAM technique must be capable of modeling the environment at any object level, most especially in human predominant environment. Furthermore, environmental noise (shadow and light intensity), sensors noise, gaussian noise etc need to be eliminated to attain a SLAM technique with maximum accuracy, even though the review provided some isues that need to be resolved to present the future SLAM. However, all problems need to be accomplished in real-time. Therefore, the research direction for this study, given the proposed DIK-SLAM, will consider and address all the above-mentioned problems.

3.4 Chapter summary

This chapter focused on the techniques used by previous researchers to address the problem of SLAM. It emphasised on their system performance, methods, advantages and limitations so as to accomplish the following:

- To understand the trend of problems affecting SLAM to date.
- To describe how the proposed research is related to prior researches in statistics
- To support the originality and relevance of the research problem
- To justify the proposed methodology and demonstrate the readiness to complete the research.

CHAPTER FOUR

4 METHODOLOGY

This section presents the description of the DIK-SLAM technique used to address the problem of SLAM taking into consideration the processing time. In developing an effective SLAM technique, this chapter explains fully, the component, assumption, precaution and laws that are employed to develop the SLAM technique.

The DIK-SLAM developed is implemented by using Matlab. Matlab is a fourth-generation programming language capable of implementing different numerical computing environments (Teng, 2000), and it was employed because of its ability to handle image processing and control (Teng, 2000).

4.1 The DIK-SLAM Technique

The proposed Simultaneous Localization and Mapping (SLAM) technique has five stages: Image acquisition stage, feature extraction stage, filtering stage, Simultaneous Localization and Mapping Stage and Navigation stage. In Image acquisition stage, sensors are used to collect data from the environment. In the Feature extraction stage, the module at this stage extracts features or landmarks that will be used for further processing the other stages. The Filtering stage detects and removes environmental noise (shadow and light intensity) while at the Simultaneous Localization and Mapping (SLAM) stage, the extracted landmarks are used for map creation. The Navigation stage only controls the movement of the robot by sending signal values to the actuator. Figure 4-1 shows the modified Monte-Carlo algorithm model and the operational flows of the proposed Simultaneous Localization and Mapping (SLAM) technique derived from the study of (Fernández-Madrigal and Claraco, 2013).

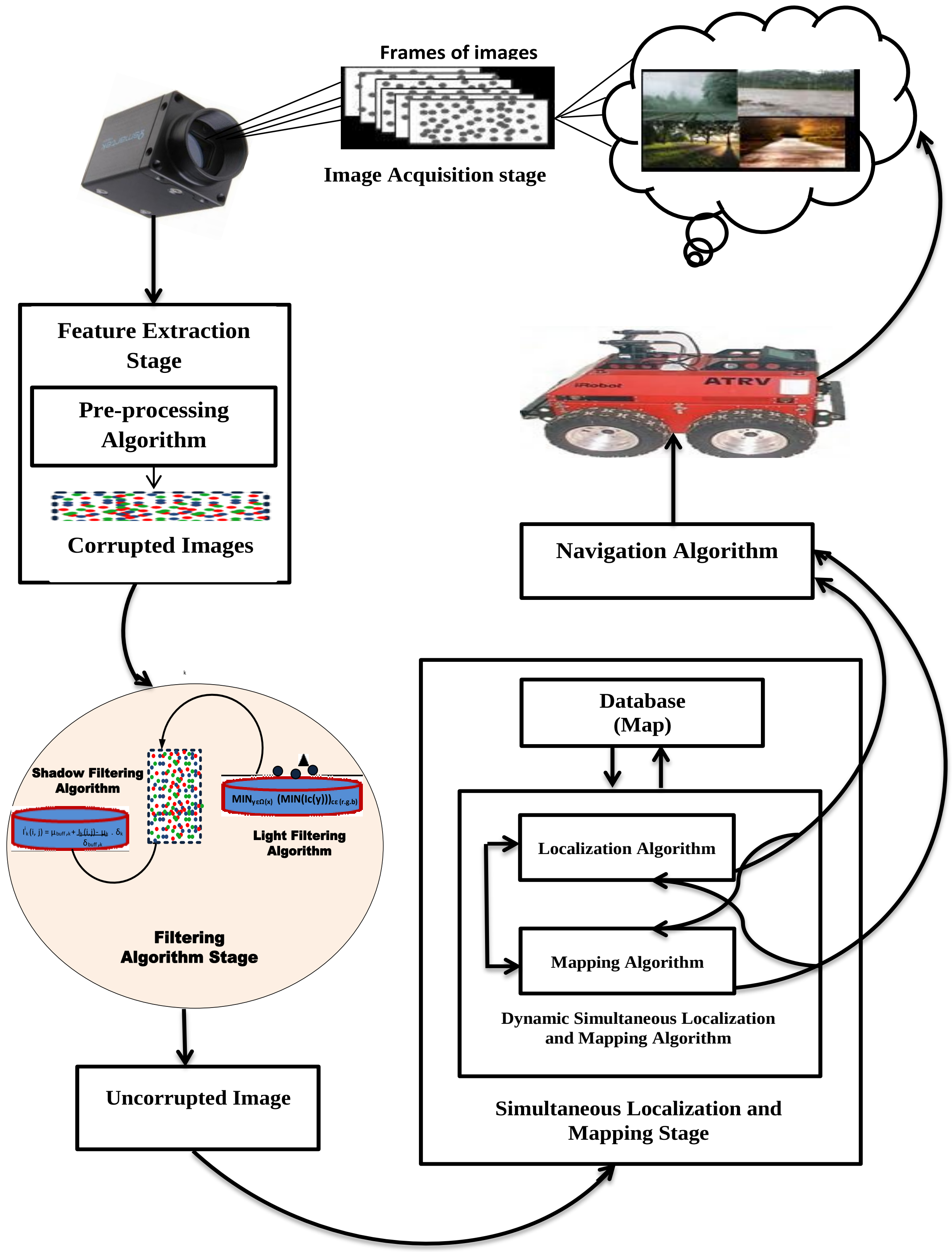

Figure 4-1: The modified Monte-Carlo algorithm model

4.2 Image acquisition stage

Image acquisition stage as related to image processing, transforms visual data captured to a continuous electrical signal that is readable (Al-amri et al., 2010)). Digital images are generated from several sensors like range sensors, radar, tomography and cameras, etc. In our technique of converting real-life images into a manageable entity, the camera sensor was employed in this stage. The camera sensor was chosen because of its advantages of gathering more information than any other sensors that will further assist in the detection process (Wen et al., 2008). The camera captured streams of images and sends to Simultaneous Localization and Mapping (SLAM) technique to process and build a map that will be used for the robot navigation (Luis et al., 2010). The image acquisition stage is the main source for which both environmental noises and dynamic landmarks appear in the image. Figure 4-2 shows the illustration of the image acquisition stage.

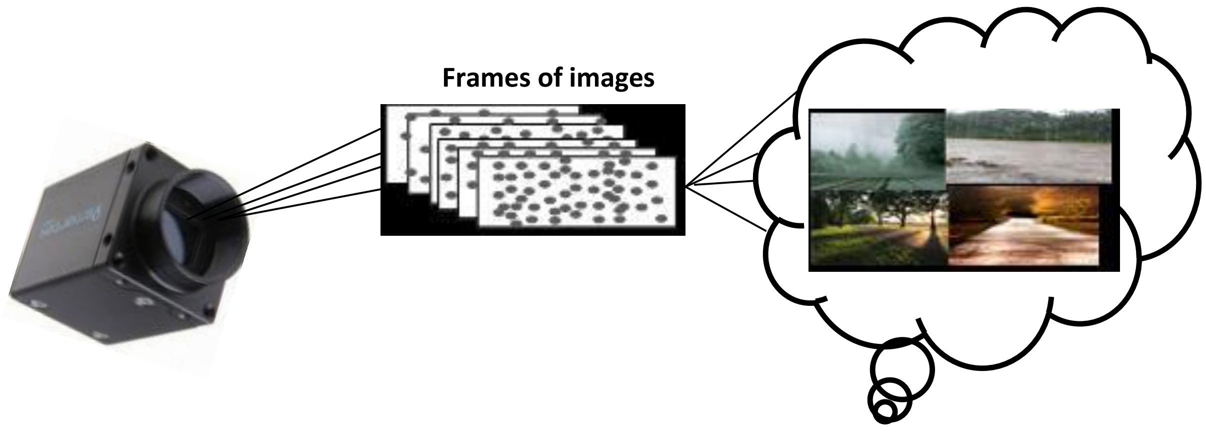

Figure 4-2: Image acquisition stage of DIK-SLAM technique

4.3 Feature extraction

The Feature extraction phase is the second stage for our road region detection system. This is a function that extracts image features of different regions (road region, non-road region and uncertainty) using statistical technique and various filters (Yenikaya et al., 2013). Image feature for Simultaneous Localization and Mapping (SLAM) technique exists in various types: colour, texture and edge, etc. In MCL-based SLAM, it is impossible for the robot to know its exact coordinate and direction in the given map, unless the robot extracts information from its environment (Luis et al., 2010). This extracted state is known as belief as expressed in equation 4-1 (Bukhori and Ismail, 2017).

$$bel(s_t) = p(s_t \setminus z_t, u_t) \qquad 4\text{-}1$$

where z_t represents the sensor measurement, u_t signifies control state while s_t represents the state sample at a time t. The belief distribution becomes a powerful statistical tool to address the problem of SLAM and it is recursively calculated from measurement and control data (Bukhori and Ismail, 2017).

In Monte Carlo localization, subsequent belief is represented by a set of samples. The expression is given in equation 3-15 (Bukhori and Ismail, 2017). Samples are a hypothesis for state representation such as wall and obstacle boundaries are orthogonal and this was proposed by (Jean-Arcady and David, 2003). These sets of samples are used to extract features that will guide the robot trajectory. Several hypotheses exist but they are not limited to the two provided above. However, some samples in the presence of environmental noise violate the hypothesis of state representation, making interpretation for the image to be impossible and these can lead to system failure (Agunbiade et al., 2013).

Algorithm 1: Feature extraction phase for the proposed SLAM technique

Input: M = frame of images from stream $1,\dots\dots\dots,m$

Output: Samples of feature extracted from M .

Step 1: Select a frame M from stream $1,\dots\dots\dots,m$ images.

Step 2: Apply equation 3-15 on M for sample extraction.

Step 3: Apply equation 4-1 on M.

Step 4: Set of Extracted features from M for every stream $1,\ldots\ldots m$ images

In algorithm 1 frame M signifies a block size of the image captured by the camera; these images contained several sets of samples appearing in various regions of the image. These samples could be used for image analysis that could solve the problem of SLAM. Figure 4-3 represents the feature extraction of the road region detection system.

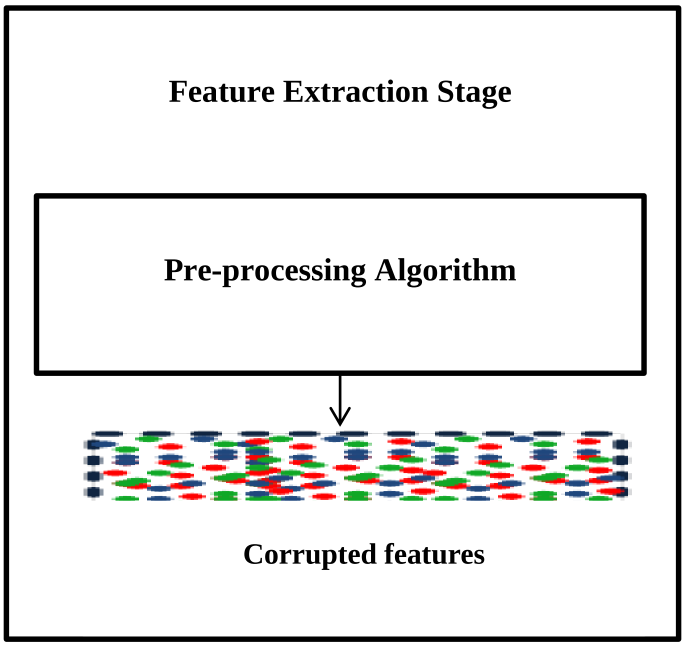

Figure 4-3: Feature extraction stage of DIK-SLAM technique

4.4 Filtering Stage

The first modification to present DIK-SLAM is to enhance its ability to overcome environmental noises that could corrupt feature properties with an effect that can lead to error pose estimation, and in worst cases, can lead to kidnapping without recovery in the case of increase in measurement noise (Negenborn et al., 2003). Environmental noises have the ability to degrade the image, bring poor vision and corrupt the RGB colour component valve, which results in poor extraction of road and non-road feature extraction. The Filtering algorithm stage is the third phase of the SLAM technique and was introduced to the system to suppress the effect of environmental noise affecting the feature extraction stage for better classification results to be achieved at SLAM phase. Two Filtering algorithms were introduced to detect and remove the environmental noises such as light intensity and shadow, because they are commonly encountered in most occasions. These filters are discussed in sections 4.4.1 and 4.4.2.

4.4.1 Shadow filtering algorithm

The Shadow algorithm is capable of suppressing the effect of shadow in an image. The functionality operation of this algorithm is based on morphological operation and normalized differences index. At first, the image RGB valve is converted to HSV because a shadow holds easy identification with the maximum value of saturation (S) and the minimum Value (V), in HSV colour space. Equation 4-2 – 4-4 expresses RGB conversion to HSV (Huang et al., 2007).

$$V = \frac{1}{3}(R+G+B) \qquad 4\text{-}2$$

$$S = 1 - \frac{3}{(R+G+B)}\min(R,G,B) \qquad 4\text{-}3$$

$$H = \begin{cases} \theta & \text{if B} \le \text{G} \\ 360^o & \text{if B} > \text{G} \end{cases} \qquad 4\text{-}4$$

where

$$\theta = \cos^{-1}\left\{ \frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}} \right\},$$

RGB is characterized as red, green, blue. The saturation (S) and value (V) components of I are used to extract the shadow area. Equation 4-5 shows illustration using Normalized Differences Index (NDI).

$$NDI = \frac{S - V}{S + V} \qquad 4\text{-}5$$

Normalized differences index images are segmented using OTSU thresholding algorithm to find an optimal threshold (T) (Krishna Kant Singh et al., 2012). The illustration is expressed in equation 4-6.

$$NDI(T) = \frac{\overline{\mu} \cdot w(T) - \mu(T)^2}{w(T).\mu(T)} \qquad 4\text{-}6$$

where is $w(T) = \sum_i^T op_i$, $\mu(T) = \sum_{i=T+2p_i}^{225}$, $\sum_{i=oi \cdot p_i}^{225}$, and p_i is the probability of pixel with gray-level I in the image [12]. Image pixels with higher NDI than the threshold (T) are classified as shadow pixels else non-shadow as expressed in equation 4-7.

$$I_{shadow}(i, j) = \begin{cases} 1 & NDI(i, j) \geq T \\ 0 & NDI(i, j) \langle \mathrm{T} \end{cases} \qquad 4\text{-}7$$

After thresholding, $\mathrm{I}_{\mathrm{shadow}}(\mathrm{i}, \mathrm{j})$ denoted as a binary image with a pixel of the shadow set to 1, and the pixel of non-shadow is set to 0. In the shadow removal process, the connected component algorithm is used for the connecting regions classified as 1. I_m Signifies connected component of I_{shadow}, illustration expressed in equation 4-8.

$$I_m = (I_{m-1} \oplus \mathrm{B}) \cap I_{shadow}, \qquad m = 1, 2, 3, ... \qquad 4\text{-}8$$

where B denotes structure element that ends operation when $I_m = I_{m-1}$. Equation 4-7 creates many ‘n’ components of the shadow area that exists in the image. The next phase is the estimation of the buffer area $\left(I_{buff,\mathrm{k}}\right)$ characterized as the non-shadow area around the shadow area. An expression illustrated in equation 4-9.

$$I_{buff,\mathrm{k}} = \left(I_{dilated,\mathrm{k}} - I_k\right) \qquad 4\text{-}9$$

where

$$I_{dilated,\mathrm{k}} = \left(I_k \oplus \mathbf{B}_{square}\right),$$

$I_{dilated,\mathrm{k}}$ denotes image dilation operation that will expand the shadow boundaries, I_k represents shadow pixel location in the image, $\mathbf{B}_{square}$ signifies 3x3 square structure element used for image, where k= 1, 2, 3, 4……….n.

In the shadow removal process, the transformation function represented as $I_k^i(i, j)$ in equation 4-10 is the mean and variance of the buffer area used to compensate the shadow regions.

$$I_k^i(i, j) = \mu_{\mathrm{buff,\ k}} + \frac{I_k(i, j) - \mu_K}{\sigma_{buff,k}} \qquad 4\text{-}10$$

where $\mu_{buff,K}$ and $\sigma_{buff,K}$ are the mean and variance of the pixels of an image I at a location $I_{buff,K}$. μ_K and σ_K are the mean and variance of the shadow pixels image I at a location I_K (Krishna Kant Singh et al., 2012).

4.4.2 Light filtering algorithm

The light filtering algorithm is capable of addressing the effect of light intensity caused by sunlight and this is a common environmental noise because of the high intensity generated from the sun. The removal effect of light intensity is estimated by modelling first object reflected by colour camera based on the dichromatic reflection model, which is the linear combination of specular and diffuse reflection components represented as $I(x)$ is expressed in equation 4-11.

$$I(x) = I^D(x) + I^s(x) = w_d(x)B(x) + w_s(x)G \qquad 4\text{-}11$$

where, *I* represents the observed image intensity, $x = \{x, y\}$ denotes the image coordinate, I^D signifies diffuse reflection component, I^S represents specular reflection component, $B(x)$ symbolizes the diffuse colour, G denotes specular colour, $w_d(x)$ signifies the coefficient that governs the magnitude of diffuse reflection component and $w_s(x)$ denotes the coefficient that governs the magnitude of specular reflection component.

The light intensity detection methodology proposed is based on the dark channel $\left(I^{dark}(x)\right)$ expressed in equation 4-12 and optimal automatic thresholding expressed in equation 4-13 to find high light reflection in an input image. The combination of this technique is used for proper classification of the area of the image affected by light intensity. The concept of light intensity detection is based on areas with affected high light, will be having high-intensity value while areas that are not affected will have low-intensity value in the dark channel model (Zou et al., 2013).

$$I^{dark}(x) = \min_{y \in v(x)}\left(\min_{c \in (r,g,b)}\left(I^c(y)\right)\right) \qquad 4\text{-}12$$

where $\nu(x)$ denotes the local patch centered at *x*, *x* represents the image coordinate and I^c signifies colour channel of *I*.

The optimal automatic thresholding (t^*) method used for dark channel image segmentation is known as valley-emphasis, it is an improved version of OSTU automatic thresholding, and it is used for labelling the dark channel image.

$$t^* = Arg\max\{(1-p_t)(\omega_1(t)\mu_1^2(t)+\omega_2(t)\mu_2^2(t))\} \qquad 4\text{-}13$$

where the threshold value is donated as t, p_t representing the probability of occurrence at the threshold value of t. μ_1 and μ_2 signifying mean gray-level of two classes, ω_1 and ω_2 representing the probability of two classes.

The marked image (x) generated by optimal automatic thresholding of dark channel image is labelled as 1, which represents the area affected by light intensity and 0 to represent areas that are not affected. Illustration thereof is expressed in equation 4-14. t^* represents the optimal threshold for image classification of mark image (x).

$$\text{mark}(x) = \begin{cases} 1 & if\, I^{dark}\ (x) \rangle\ t^* \\ 0 & Otherwise \end{cases} \qquad 4\text{-}14$$

The light intensity removal process is based on specular-to-diffuse proposed by (Zou et al., 2013). Illustration of an image without light intensity effect (I^D) is expressed in equation 4-15

$$I^D(\wedge_{\max}) = I - \frac{\max_{u\in(r,g,b)} I_u - \wedge_{\max}\sum_{u\in(r,g,b)} I_u}{1-3\wedge_{\max}} \qquad 4\text{-}15$$

However, introducing these filtering algorithms leads to high computation cost (Shengyan and Karl, 2010) and these limitations will be addressed by a concurrency filtering operation known as Rotating Tilling (RT) technique, which will be discussed in section 4.5.2.1. This technique is used to increase the processing speed of the filtering operation since the computational cost is taken into consideration (Lin et al., 2003).

4.4.3 Concurrency technique (rotating tilling)

Concurrency technique proposed for the filtering algorithms allows multiprocessing within a limited time (Chin et al., 2001). Concurrency technique was invented to overcome the limitation of the traditional method (serial operation) with delay processing speed. Various techniques of concurrency exist, but the two well-known methods are Parallel Pipelined (PP) and Binary Swapping (BS) technique (Lin et al., 2003). In Parallel Pipelining, ring rotation topology is employed for partial image composition. This allows arbitrary numbers of processors to use but with larger communication steps. In Binary Swapping, this technique can only function when the number of processors is limited to a power of 2 with less communication step than the parallel pipeline technique (Lin et al., 2003). In this research, we employed the use of Rotating Tilling (RT) (Lin et al., 2003). It combines the advantages for both Parallel Pipelined and Binary Swapping techniques to overcome their limitations. This was possible because the processor arrangement functions on ring rotation topology (parallel pipelined) and the data communication technique is based on indexing the operation of the Binary Swapping method. In Rotating Tilling technique, three stages are involved during the processing of the image and are discussed in section 4.4.3.1 - 4.4.3.3

4.4.3.1 Image (data) partitioning stage

The objective at this phase is to distribute image volume across processors and to minimise computation cost (Lin et al., 2003). In the RT technique at the data partitioning stage, any efficient data partitioning technique can be employed and various types of this technique exist (Lin et al., 2003). In (Sano et al., 2000) slice data partitioning distributes evenly, volume image slices to processor but this result into high overhead communication and excess computation time at the image compositing phase while in (Zhang et al., 2005) volume data partitioning revealed, has similar characteristics with even volume distribution across processors with minimised overhead communication and excess computation time at the image compositing phase. The shared volume data partitioning mentioned in (Zhang et al., 2005) minimises the overhead communication with less computation time at the image compositing phase, but image volumes are not evenly distributed across processors.

4.4.3.2 Image (data) rendering stage

This is the second phase after the operation of the volume data (image) partitioning, which resulted in the partial image depending on the number of processors (Sano et al., 2000). Each partial image assigned to one processor applies the rendering operation. The rendering algorithm performs encoding, resampling and generation of the corresponding initial block from the partial image (Lin et al., 2003). In the operation, block images belonging to each processor are numbered before distribution across corresponding processors and can avoid overlapping because initial blocks are created independently by each processor (Lin et al., 2003). The distribution allows the simultaneous detection and removal process of the environmental noises. In the RT technique, using the processor (P_r), the block size $\left(A_r^k(m)\right)$ of its partial image is sent to corresponding processors $\left(P_i\right)$ using equation 4-16 and receives block size $\left(A_j^k(m)\right)$ of a partial image from processor $\left(P_j\right)$ using equation 4-17 (Lin et al., 2003).

$$P_r\left(A_r^k(m) \rightarrow P_1\right) where \begin{cases} l = 0,1.....,k-1 \\ w = 0,1.....,\lceil P/N \rceil - 1 \\ v = 0,1,....,\lceil P/2^k \rceil \\ m = \left(\left(r - 2^{k-1} + 2l\right) \bmod 2^k\right) 2^{k-1} + 2^{2k-1}v + P \\ i = (r - 2^{k-1} + l) \bmod P \end{cases} \qquad 4\text{-}16$$

$$P_r \leftarrow P_j\left(A_j^k(n)\right), where \begin{cases} l = 0,1.....,k-1 \\ w = 0,1.....,\lceil N/P \rceil - 1 \\ v = 0,1,....,\lceil P/2^k \rceil \\ m = \left((r+l) \bmod 2^k\right)2^{k-1} + 2^{2k-1}v + Pw + 2l \\ i = (r - 2^{k-1} - l) \bmod P \end{cases} \qquad 4\text{-}17$$

where, j, r and i represent processor ranks, k is a positive integer while m and n are represented as blocks numbers.

4.4.3.3 Image (data) composition stage.

This is the last phase, after the rendering operation, which resulted in a number of initial blocks. At this stage, each processor composites the block it received during distribution using the Over-operation. MPICH, a gather directive of a message passing library on multicomputer memory distribution, is used for merging this block until the final image (filtered image) is generated (Lin et al., 2003). This is achieved in a short period of time with less environmental effects. In the Rotating Tilling technique, $\lceil \log P \rceil$ communication steps exist, when the communication step $K = 1$, $\left\lceil \frac{N}{P} \right\rceil$ is represented by a maximum number of send/receive operation performed by processors and $\frac{A}{N}$ is denoted as the block size sent or received by a processor. The output result of maximum data communication $\left(T_{comm}RT\right)$ and the computational time $\left(T_{comp}RT\right)$ among the processors is expressed below in equation 4-18 and 4-19.

$$T_{comm}RT = \left\lceil \frac{N}{P} \right\rceil * T_s + \left\lceil \frac{N}{P} \right\rceil \frac{A}{N} * T_p \qquad 4\text{-}18$$

$$T_{comp}RT = \left\lceil \frac{N}{P} \right\rceil \frac{A}{N} * T_o \qquad 4\text{-}19$$

When the communication step (K) is more than 1, $\left\lceil \frac{2B_k}{P} \right\rceil$ represents a maximum number of send and receive operations performed by processors. $\frac{A}{N \cdot 2^{K-1}}$, $\left(K = 2,....,\lceil \log P \rceil\right)$ denotes block size for a send and receive operation by each processor,

where $$B_K = \begin{cases} N & \text{for } k = 1 \\ B_{k-1-\lfloor B_{K-1}/P \rfloor} & \text{for } k \rangle 1 \end{cases} \qquad 4\text{-}20$$

The output result for maximum data communication time $(T_{comm}RT)$ and computational time $(T_{comp}RT)$ among processors is illustrated below in equation 4-21 and 4-22

$$T_{comm}RT = \left\lceil \frac{2B_k}{P} \right\rceil T_s + \left\lceil \frac{2B_k}{P} \right\rceil \frac{\mathrm{A}}{\mathrm{N} \cdot 2^{\mathrm{K}-1}} T_p \qquad 4\text{-}21$$

and

$$T_{comp}RT = \left\lceil \frac{2B_k}{P} \right\rceil \frac{\mathrm{A}}{\mathrm{N} \cdot 2^{\mathrm{K}-1}} T_o \qquad 4\text{-}22$$

The total processing time represented as $T_{total}(RT)$ according to the above equations is formulated in equation 4-23.

$$T_{total}(RT) = \left(\left\lceil \frac{N}{P} \right\rceil + \sum_{K=2}^{\lceil lop\, P \rceil} \left\lceil \frac{2B_K}{P} \right\rceil \right) T_s + \left(\left\lceil \frac{N}{P} \right\rceil \frac{A}{N} + \sum_{K=2}^{\lceil \log P \rceil} \left\lceil \frac{2B_K}{P} \right\rceil \frac{A}{N 2^{k-1}} \right) T_p + \left(\left\lceil \frac{N}{P} \right\rceil \frac{A}{N} + \sum_{K=2}^{\lceil \log p \rceil} \left\lceil \frac{2B_K}{P} \right\rceil \frac{A}{N 2^{k-1}} \right) T_o$$

$$= \left(\sum_{K=1}^{\log P} \left\lceil \frac{2B_K}{P} \right\rceil + \left\lceil \frac{N}{P} \right\rceil - \left\lceil \frac{2N}{P} \right\rceil \right) T_s + \frac{A}{N} \left(\sum_{K=1}^{\lceil \log p \rceil} \left\lceil \frac{2B_K}{P} \right\rceil \frac{1}{2^{k-1}} + \left\lceil \frac{N}{P} \right\rceil - \left\lceil \frac{2N}{p} \right\rceil \right) \left(T_p + T_o \right) \qquad 4\text{-}23$$

where $T_{total}(RT)$ is factorized by the data transmission time per byte T_p the computation of the over operation per pixel T_0, the image size in pixels A, the start-up time of a communication channel T_s, σ represents a total maximum amount of data exchange between processors, the number of initial block of partial image N and the number of processors P (Lin et al., 2003).

Algorithm 2: Filtering stage for the proposed SLAM technique

```
Input: $P$ represents numbers of processor
       $A$ represents numbers of partial image size
       $N$ represents numbers of an initial block from partial images
       $K$ represents the communication step
       $M$ represents the frame of images from stream $1.....m$
Output: Filtered Image $M$
Step 1: Data distribution is applied to $M$ generate $A$
Step 2: Processors break $A$ into $N$ block of images
Step 3: Each processor in the ring rotation topology sends $N$ block of its images across to
corresponding processors
Step 4: the Corresponding processor receives $N$ block of images from other processors
Step 5: Each processor composites the block received with its local block using Over-
operation.
Step 6: Processors in the ring rotation topology that sent $N$ block of images to others and
idle are removed from the ring.
Step 7: Other processors in the ring rotation topology break composite image into two equal
halves
Step 8: Repeat steps 3-7 for several $k$ until the final image is generated.
```

In algorithm 2, the concurrency method used allows the frame M receives high computing performance over a short period of time because simultaneous filtering operations are carried out on M (Chin et al., 2001). Figure 4-4 illustrates filtering algorithms in concurrent operation. This stage is the first algorithm introduced for the re-modification of DIK-SLAM and the output here is an image with a higher quality image ,which allows easy recognition of samples that will be employed to localization and mapping at the next phase.

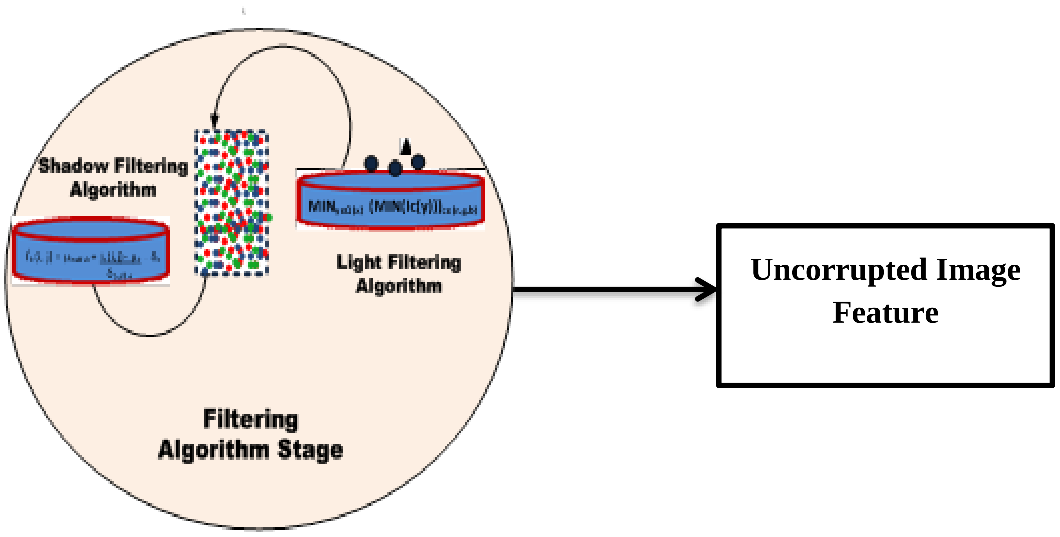
Shadow Filtering Algorithm
Light Filtering Algorithm
Filtering Algorithm Stage
Uncorrupted Image Feature

Figure 4-4: Filtering stage of DIK-SLAM technique

4.5 Simultaneous Localization and Mapping stage

Localization is the ability of a robot to establish its own orientation and position within an environment, while mapping is the ability of a robot to construct a safe path for the unknown environment (Oh et al., 2013). These operations are fundamental problems for mobile robots. The problem of localizing a robot and building a map in robotics is a different task but deeply associated with each other because the mobile robot needs the map to localize itself and the accurate position of the robot is needed to build the map (Oh et al., 2013). In an attempt to solve this interconnection between the two problems, Simultaneous Localization and Mapping (SLAM) technique has attracted many researchers towards focusing on this area and many

Simultaneous Localization and Mapping (SLAM) techniques have been proposed in the literature with outstanding result attained, but in static environments (Fuentes-Pacheco J, 2015, Clipp et al., 2009) But in this study, we improved the Simultaneous Localization and Mapping (SLAM) technique to cope with both static/dynamic environments. However, the database introduced into the SLAM technique shown in the Figure 4-1 will assist to carry out Initializing localization, which could be used to address Kidnapped robot. At this stage, the original Monte-Carlo algorithm presented on pages 23-25 was modified to solve the problem of SLAM and will be discussed in section 4.5.1.

4.5.1 The Second Re-modification in DIK-SLAM

The DIK-SLAM model relies on the Monte-Carlo algorithm for localization and map estimation, but this algorithm must be enhanced to cope with dynamic, kidnapping and loop closure, since there is a possibility of encountering them. Monte-Carlo algorithm is a common probabilistic algorithm for solving the issue of concurrent localization and mapping. This technique has produced satisfactory results, unlike other algorithms such as EKF and Kalman filters, that suffer from high computational complexity (Shiguang et al., 2017). This limitation has shifted researchers attention towards particle-based algorithm (Abouzahir et al., 2014). Thus, in this research, high computational complexity is a problem we are considering to minimize and lead to proposing a particle-based technique known as Monte-Carlo algorithm but limited to the Static environment (Thrun et al., 2001). In an environment that is dynamic in nature, the Monte-Carlo algorithm needs to be upgraded to cope with such an environment. In this research, the second modification to present DIK-SLAM ability to cope with dynamic environments only will be discussed in this section.

The dynamic environment has the ability to change because of the presence of moving objects and to handle such condition, the technique relies on altering the map as changes happen in the environment. In the modified MCL each cell of the map is considered to be an independent object as expressed in equation 4-24

$$Y_t = \left(y_{1,t}, \ldots\ldots\ldots, y_{k,t}\right) \qquad \text{4-24}$$

Where y_t signifies a set of individual cells in the map.

Based on the independent cell assumption, the new state equation $p\left(y_t, x_t \middle| z_t, u_t\right)$ will be factorized to add a new probability for the cells in the map to the original MCL algorithm discussed in chapter 3. This factorization technique is adopted and similar to the technique proposed in (Avots et al., 2002) for adding the state of a door into the MCL algorithm. In this factorization, the Bayes rule and the Markovian presented in equation 4-25 are the basis computation that needs to be factorised to generate the new state.

$$p\left(y_t, x_t \middle| z_t, u_t\right) = \eta p\left(z_t, x_t\right) p\left(y_t \middle| x_t, z_{t-1}, u_t\right) p\left(x_t \middle| z_{t-1}, u^t\right) \qquad \text{4-25}$$

The state-space $\left(y_t, x_t\right)$ in equation 4-25 is exponential to y_t in size and must be resolved by reducing its state space to achieve the goal. Given that the state variable is equally likely and the probability for a random sensor scan is constant. Hence,

$$p\left(z_t \middle| x_t, y_t\right) = \frac{p\left(x_t, y_t \middle| z_t\right) p\left(z_t\right)}{p\left(y_t, x_t\right)} = \eta^! p\left(x_t \middle| z_t\right) \prod_k p\left(y_{k,t} \middle| x_t, z_t\right) \qquad \text{4-26}$$

Taking into consideration that those cells in the map can independently change status in the model and re-applying the Markovian assumption equation 4-27 was derived.

$$p\left(y_t \middle| x_t, z_{t-1}, u_t\right) = \prod_k \sum_{y_{k,t-1}} p\left(y_{k,t} \middle| y_{k,t-1}\right) p\left(y_{k,t-1} \middle| x_{t-1}, z_{t-1}, u_{t-1}\right) \qquad \text{4-27}$$

Finally:

$$p\left(x_t \middle| z_{t-1}, u_t\right) = p\left(x_t \middle| x_{t-1}, u_t\right) p\left(x_{t-i}, \middle| z_{t-1}, u_{t-1}\right) \qquad \text{4-28}$$

Recombining and simplifying equation 4-26, 4-27 and 4-28, the factorization result is provided in equation 4-29

$$p(y_t, x_t | z_t, u_t) = p(x_t | z_t, u_t) \prod_k p(y_{k,t} | x_t, z_t, u^t) \qquad 4\text{-}29$$

Which contains the addition of a new probability for the cells in the map to the original MCL posterior, which plays a role addressing moving object but calculating the new probability for the cell, will be discussed in section 4.5.1.1. Thus, detail factorization of adding the new probability of the cells to MCL can be seen in the work of (Avots et al., 2002).

4.5.1.1 Bayes Filtering of Binary Object in an environment

MCL is referred to as a recursive Bayes filter that has been modified to cope with new probability for the cell on the map. However, the study hasn't discussed how to calculate the probability of each cell in the map, which plays a significant role in dealing with moving objects. In this research, the method of calculating cell probability is similar to the work proposed by (Avots et al., 2002). In this technique, these cells are a binary object which can either be absent or present if occupied by a real object. Therefore, map cells can either be 0 or 1 with a probability summation equal to 1 as defined as $\pi_{k,t}$ which represents a single numerical probability:

Let: $\pi_{k,t} = p(y_{k,t} = 1 | x_t, z_t, u_t)$ 4-30

Then

$$\pi_{k,t} = \frac{p(y_{k,t} = 1 | x_t, z_t) p(z_t | x_t)}{(y_{k,t} = 1) p(z_t | x_t, z_{t-1}, u_t)} \pi^{+}_{k,t}$$

Where

$$\pi^{+}_{k,t} = p(y_{k,t} = 1 | y_{k,t-1} = 1) \pi_{k,t-1} + p(y_{k,t} = 1 | y_{k,t-1} = 0)[1 - \pi_{k,t-1}]$$

But when $y_{k,t} = 0$, when the cell is not occupied, represent an opposite event of equation 4-30.

The probability of such an event is $1 - \pi_{k,t}$:

$$1-\pi_{k,t} = \frac{p(y_{k,t}=0|x_t,z_t)p(z_t|x_t)}{p(y_{k,t}=0)p(z_t|x_t,z_{t-1},u_t)}\pi_{k,t}^{-} \qquad 4\text{-}31$$

Where

$$\pi_{k,t}^{-} = p(y_{k,t}=o|y_{k,t-1}=1)\pi_{k,t-1} + p(y_{k,t}=0|y_{k,t-1}=0)[1-\pi_{k,t-1}]$$

Given the unknown quantities in equation 4-30 and 4-31, the two are divided to cancel the unknown quantities as expressed in equation 4-32 with known quantities but commonly referred to as odd ratio

$$\frac{\pi_{k,t}}{1-\pi_{k,t}} = \frac{p(y_{k,t}=1|x_t,z_t)}{1-p(y_{k,t}=1|x_t,z_t)}\frac{1-p(y_{k,t}=1)}{p(y_{k,t}=1)}\frac{\pi_{k,t}^{+}}{\pi_{k,t}^{-}} \qquad 4\text{-}32$$

The equation 4-32 consists of known probability where $p(y_{k,t}=1)$ represents prior probability for an occupied cell, $p(y_{k,t}|y_{k,t-1})$ signifies transition probability state for a cell, $\pi_{k,t-1}$ denotes prior occupancy probability state and $p(y_{k,t=1}|x_t,z_t)$ represents the occupancy probability of a cell given sensor data and robot location. However, the odds ratio can be resolved using an equally desired probability $\pi_{k,t}$ expressed in equation 4-33

$$\pi_{k,t} = 1-\left(1+\frac{\pi_{k,t}}{1-\pi_{k,t}}\right)^{-1} \qquad 4\text{-}33$$

In this study, the technique presented an updated map dynamically during localization to avoid increasing complexity of the system with an effect that does not severely increase the computational time.

4.5.1.2 The Third re-modification in DIK-SLAM using Similarity and Dissimilarity in an image for Resolving Kidnap problem and Loop Closure

The third re-modification to present DIK-SLAM was to upgrade again the Monte-Carlo algorithm to cope with the problem of Kidnapping. The kidnap problem in SLAM happens when an unexpected movement occurs to the robot in an environment. This issue arises during a failing sensor or intensification in measurement noise (Guyonneau et al., 2012.). The effect of this limitation leads to a robot inability to estimate its pose, therefore, violating the principle of addressing the problem of SLAM because pose estimation is a requirement for map building, can lead to robot failure without recovery (Sounderhauf et al., 2012). The algorithm selected to solve this problem must be able to carry out these objectives: the ability to perform pose estimation, detect kidnap problems and global localization (Guyonneau et al., 2012.). In the literature, the scan to map matching is a common technique to address kidnapping. In this technique, the matching algorithm is introduced for an extensive search of the current observation over the reference map (pre-mapped environment) to extract robot position in relation to the selected reference map as proposed in the work (Se et al., 2001, Matei et al., 2013). However, there is a situation where this technique can experience failure. Take for instance, if the current observation has changed because of dynamic characteristics, which might be impossible to select a perfect match of a reference map and could lead to SLAM failure without recovery (Tian and Ma, 2017). In this study, the proposed DIK-SLAM technique will be re-modified for the second time to minimize this limitation. This technique will take into consideration similarities and dissimilarities between the reference image and current observation image before performing its re-localization procedure. Given an autonomous robot navigating in an unknown environment using the DIK-SLAM technique, how would we know that the robot is kidnapped? Once kidnapping occurs, particle migrate from local sample to global samples and after re-localization global samples are migrated has local samples. The global samples are a key feature towards robot kidnapping and recovery. Furthermore, the particle probabilities are taken into consideration. If the maximum of probability of the particle is less than the sensitive coefficient γ the robot triggers kidnapping. This technique was adopted by the work of (Chuho and Byung-Uk, 2011), although their technique relies on entropy value of particles while the DIK-SLAM technique relies on the weight of the particles. Expression is given in equation 4-34

$$Kidnapped\ robot_t = \begin{cases} 1 & w_t^{\max} > \gamma \\ 0 & Otherwise \end{cases} \qquad 4\text{-}34$$

In kidnap conditions, the scan to match algorithm considering their appearance relies on the SIFT descriptor to match the reference image of the current view to a previously built map that will be used for the re-localization purposes. This technique was proposed by (Se et al., 2001) and was adopted because of its outstanding performance. However, the Monte-Carlo algorithm was re-modified again by introducing the Fourier signature of the similarity and dissimilarity function as illustrated in 4-35 and 4-36 respectively for measurement purposes in images (Ishiguro and Tsuji, 1996)

$$Disim(I_i, I_t) = \sum_{y=0}^{l-1} \sum_{k=0}^{m-1} \left| F_{iy}(k) - F_{ty}(k) \right| \qquad 4\text{-}35$$

$$Sim(I_i, I_t) = 1000 - 1000 \frac{Disim(I_i, I_t) - Min_{it}\{Disim(I_i, I_t)\}}{Max_{it}\{Disim(I_i, I_t)\} - Min_{it}\{Disim(I_i, I_t)\}} \qquad 4\text{-}36$$

where I_i and I_t represent respectively the reference image and current observation image, while $F_{iy}(k)$ and $F_{ty}(k)$ signify the Fourier coefficients of the k^{th} frequency of yth row image in the reference image and current observation image.

The higher the value of the similarity function, the more similar the reference image and the current observation image (Ishiguro and Tsuji, 1996). Given this condition, the matching map of the reference image can be used to estimate the robot position.

However, if the dissimilarity function is high, the more dissimilar the reference and current observation image (Ishiguro and Tsuji, 1996). In this condition, the matching map of the reference image will not be used, rather, we propose a new solution of generating a new set of samples S_t from the current observation image for the re-localization process since the dissimilarity in the reference image compared to the current observation image might not provide us with a satisfactory result. The set of new samples can be used to generate a map, which will be used to estimate a reliable robot position that will assist the robot to keep navigating successfully till it gets to a familiar environment, where it can use its reference map to continue estimating its position.

4.5.1.2.1 The Fourth Modification in DIK-SLAM for Loop Closure Detection in SLAM

The Fourth re-modification to present DIK-SLAM was to upgrade again the Monte-Carlo algorithm ability to cope with the problem of loop closure. The loop closing problem in SLAM is a task for determining if an autonomous vehicle during exploration has returned to a formerly visited environment (Newman and Kin, 2005). The loop closure is comparable to kidnapping, expect that the robot is aware of its environment, but needs to know if the environment has been re-visited (Agha-mohammadi et al., 2015). However, in appearance-based localization technique, the underlying principle is, if the current image matches one of the reference images stored in the database using the SIFT descriptor (built from previous navigation), then a robot has returned to its previous location and the loop must be closed otherwise, kidnapping is triggered (Ho and Newman, 2007). Considering the problem of loop closure, the MCL based Bayesian algorithm was re-modified to address this issue. Given that a robot re-visited an environment, the full posterior must be estimated and to find if Loop-Closure occurred, the Bayes rule under the Markov assumption can be decomposed into equation 4-37

$$p(S_t | I^t) = \eta p(I_t | S_t) p(S_t | I^{t-1}) \qquad 4\text{-}37$$

Where η represents the normalization term, S_t signifies the random variables representing loop-closure hypotheses at the time t and $I^t = I_0........,I_t$ represents the sequence of images in the SIFT features space and can consequently be represented by the sequences $z^t = z_0.......,z_t$.

Therefore, the $p(S_t|I^t)$ will be replaced by $p(S_t|z^t)$ to derive the full posterior expressed in equation 4-38

$$p(S_t|z^t) = \eta p(z_t|S_t) p(S_t|z^{t-1}) \qquad 4\text{-}38$$

Where $p(z_t|S_t)$ is the likelihood $L(S_t|z_t)$ of S_t given the words z_t closes the loop with the matched previous image (Angeli et al., 2008).

Algorithm 3: Simultaneous Localization and mapping stage for the proposed SLAM technique

```
Input: $M$ = frame of filtered images from stream $1,........,m$
Output: Model representation (map) of $M$.

Step 1: Select a frame $M$ from stream $1,........,m$ images.
Step 2: Apply 4-29 on M for sample extraction.
Step 3: if kidnap is true apply equation 4-35 and 4-36
            Else
            Loop closure problem apply 4-37 and 4-38
            return to step 2
Step 4: model representation for stream $M$ for every stream $1,........,m$ images
```

In algorithm 3 frame M signifies a block size of the image captured by a camera, which is analysed by using sets of samples appearing in various region of the image to address simultaneous localization and mapping and dynamic environment, while the database will assist to address the issue of kidnap problem and loop closure. The outcome at this stage will result in a model representation of the environment captured. This stage will attain a good result because environment noises, which degrade images characteristic have been addressed at the previous stage. These environmental noises are widely complained by many researchers (Irie et al., 2012., Cras et al., 2011) however, they are not only limited to shadow and light intensity, but there are also others like rain, snow, mist, fog, humidity, etc which can be encountered in an outdoor environment. The study only targets shadow and lighting variation because they are the likely noises we can encounter in the data used for carrying out this experiment. Figure 4-5 represents the simultaneous and localization stage of the proposed SLAM technique (DIK-SLAM).

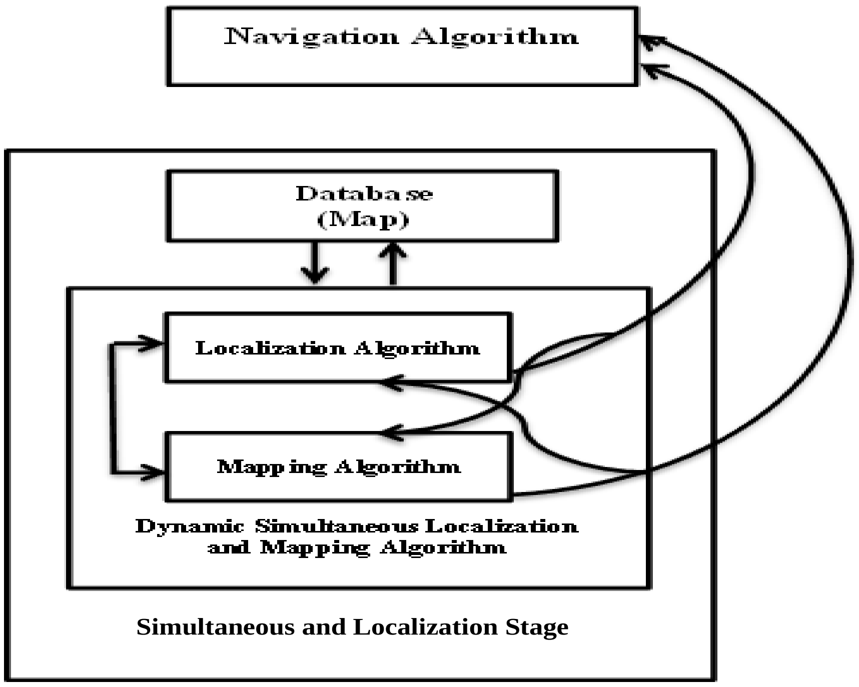
Navigation Algorithm
Database (Map)
Localization Algorithm
Mapping Algorithm
Dynamic Simultaneous Localization and Mapping Algorithm
Simultaneous and Localization Stage

Figure 4-5 Simultaneous and localization stage of DIK-SLAM technique

4.6 Navigation Algorithm stage

Autonomous robot system needs a reliable estimate for the current robot position and a precise map of the environment for perfect navigation to be attained, but this can be solved at the fourth stage (SLAM). Thus, the problem of path planning, which considers a map of an environment or a model to extract an optimal pathway for an autonomous robot to navigate from the starting position to the end position of the map, must be addressed (Guyonneau et al., 2012). This problem becomes cumbersome because the autonomous robot will not have complete map information and any pathway generated from its initial map may be unacceptable or substandard as it constantly collects updated map information from its sensor (Djekoune. et al., 2009). Therefore, it is important for the navigation algorithm of the robot to be able to update its map and re-plan its optimal pathway once new information is received from its on-board sensors. In the literature, various navigation algorithms have been proposed, but A^* and D^*algorithm is common. These two are widely used algorithms(Djekoune. et al., 2009), because of their capabilities in efficient use of heuristics and the ability to cope with increased updated map information from its sensor. However, the D* algorithm was selected in this study because it has been modified for improving its performance over the older version (Zhang and Li, 2017). Thus, this section discusses the important information on how the algorithm operates. The D* algorithm relies on a grid-based approach that split the area of exploration into $n \times n$ grid. The grid-based algorithm (D*) relies on the cost function to plan the trajectory/path. The basic cost function of the D* algorithm based on distance travelled is presented below. Let k be the cost of moving vertically or horizontally $(horizontal \mid vertical)$ in the cell, the cost function is expressed in equation 4-39 and 4-40.

$$\cos t(x, y) horizontal \mid vertical = k * Units \qquad 4\text{-}39$$

$$\cos t(x, y) diagonal = \left(\sqrt{2} * k\right) * units \qquad 4\text{-}40$$

where the Unit represents an attached weight that varies depending on the size of the cell.
The cost function in equation 4-39 and 4-40 are recomputed to include factors, the terrain slope and its azimuth (the horizontal angle). Considering a robot navigating between two points $A(x_1, y_1, z_1)$ and $B(x_2, y_2, z_{12})$, the slope is computed using equation 4-41

$$slope = \frac{(z_2 - z_1)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \qquad 4\text{-}41$$

While the azimuth (the horizontal angle (θ)) is computed using equation 4-42

$$\theta = \left(\frac{180}{\Pi}\right)\tan^{-1}\left((x_2 - x_1)/(y_2 - y_1)\right) \tag{4-42}$$

Where x, y, z represent state variables for point A and B, θ signify the horizontal angle. The re-computed cost function is given in equation 4-43

$$\cos t\_function = \cos t(x, y) + \theta(slope) * factor \tag{4-43}$$

Where $factor$ in equation varies based on different terrain. In ascending, a higher factor is given than descending and in level terrain, factors are made zero (Saranya et al., 2016). Given equations 4-43 the cost functions are calculated for all cells and the robot moves through the cells with the lowest cost functions. In D* algorithm, this is the basic concept of how the robot trajectory is created. Thus, the D* algorithm has been modified to improve its performance. In this study, the enhanced D* lite algorithm proposed by (Ganapathy et al., 2011) will be employed to implement this stage. This algorithm was selected because it can cope with dynamic mapping and most importantly it functions in real-time.

Algorithm 4: Navigation stage for the proposed SLAM technique

```
Input:  The model representation of the Environment (map) of $M$ from stream 1.....$m$
Output: Robot Trajectory

Step 1: Input map of $M$ from stream 1.....$m$
Step: 2 Apply the grid-based approach to spilt M from stream 1.....$m$ to $n \times n$ cell
Step 3: Apply the cost function in 4-43.
Step 4:  Compute Robot trajectory
```

In algorithm 4, the idea of proposing the navigation stage into the system is to assist with the computation of the robot trajectory. This algorithm has the ability to extract the safest path for a robot. Figures (4-6) illustrate the navigation stage of the proposed Simultaneous Localization and Mapping (SLAM) technique.

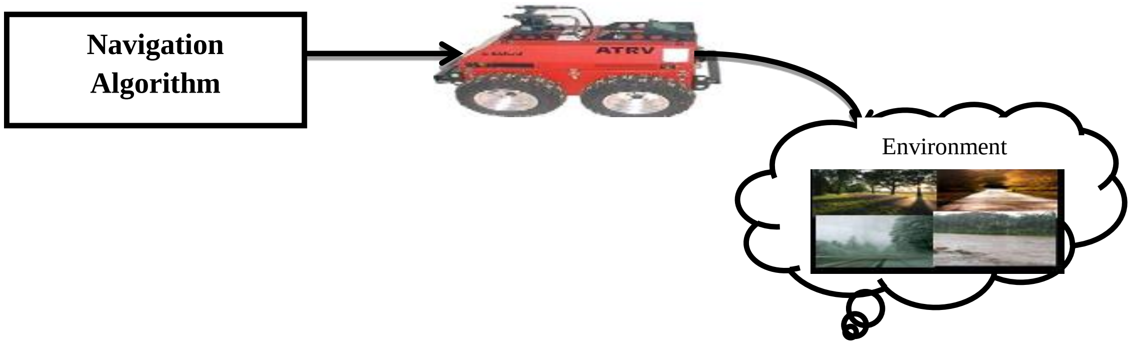

Figure 4-6 Navigation stage of DIK-SLAM technique

4.7 Computational Complexity (CC)

Computational Complexity (CC) is one of the major setbacks related to SLAM in terms of hardware cost for implementation. In order to improve this setback, particle filtering was introduced so as to reduce the number of samples that we analyze. We consider that the exact Computational Complexity (CC) will perform on every sample in order to get a projection. The combination of particles at a particular position into a single particle is referred to as particle filtering, in which which weight is allocated to that particle to reflect the number of particles that were combined to form it. This eliminates the need to perform a redundant computation without the probability distribution. In particle filtering, this is accomplished by sampling the system to create P particles, then comparing the samples with each other to generate an importance weight, thereafter, normalizing the weights, which resample P particles from the system using these weights. This process greatly reduces the number of particles that must be sampled, making the system much less computationally intensive. The states (s) to be estimated are the coordinates and orientations of the robot ,which are x, y, θ and w .

$$s = (x, y, \theta, w) \qquad 4\text{-}44$$

Where each represents x, y coordinates, phase angle and weight respectively. This defines the state of the mobile robot, which determines the motion control and path evaluation defined as:

$$\min\left(\frac{m}{f}\right)^2 \qquad 4\text{-}45$$

Where $f = 7$ and $m = \sum step - gap + 1 : step \leq \max step$, where $\max step = 1300$ in gaps of 20

The path of the robot can be calculated based on:

$start\_index = \frac{startpo\text{int} - origin}{xy}$, the x and y function, which determines the x and y index can be defined as:

$$x_{func} = startpo\text{int} + (x - startpo\text{int}) \times \frac{(endpo\text{int} - startpo\text{int})}{(endpo\text{int} - startpo\text{int})} \qquad 4\text{-}46$$

$$y_{func} = startpo\text{int} + (y - startpo\text{int}) \times \frac{(endpo\text{int} - startpo\text{int})}{(endpo\text{int} - startpo\text{int})} \qquad 4\text{-}47$$

Related to x and y respectively. A decision metric of, if the difference of end index and start index is greater than zero related to x parameters, employ the y function, and if the difference in end index and start index is greater than zero related to y, employ the x function. In the proposed SLAM technique (DIK-SLAM), the study has shown the ability to remodify the Monte-Carlo algorithm to add new observations and maintaining a minimal ancestral tree, which requires $O(AP)$ time. However, the part that intensifies the proposed technique is the filters and has been minimized by the concurrency technique. The other part of the algorithm that could increase its complexity is the operation of the resampling, which happens at the Localization and Mapping stage and not to further increase its complexity. We avoid introducing a standalone algorithm. Rather, techniques used to address the issue of kidnapping and dynamic environment are Monte-Carlo based technique with an effect that does not increase it beyond unacceptability. Therefore, for P particles, the time taken is given in equation 4-48

$$O(P \log P) \tag{4-48}$$

While the overall complexity of the proposed algorithm is given in equation 4-49

$$complexity = O(P \log P) + O(AP) \tag{4-49}$$

$$= O(P \log P + AP)$$

Thus, the area covered by the sensors matters, and could be larger than , $\log P$ which can also contribute to high computational complexity, as well as the numbers of particles used, can also contribute to high lead to high computational complexity. Given this study, we are satisfied in stating the complexity of the algorithm as simply $O(AP)$ (Eliazar and Parr, 2019).

4.8 Chapter summary

This section discussed the DIK-SLAM technique, operating based on the re-modification of the Monte-Carlo algorithm to cope with dynamic and kidnapping issues. The image acquisition stage was employed to capture lifestreams of images that are sent to the SLAM technique for processing. At the feature extraction stage, samples for different regions were used for building maps and providing guidance for robot trajectory. In the presence of environmental noises, these samples were difficult to extract. To resolve this issue, filtering algorithms working in parallel mode, were employed to minimize the effect caused by shadow and light intensity, which had assisted the mapping and localization stage to achieve better results, while assisting the navigation algorithm attain successful navigation for the robot.

CHAPTER FIVE

5 EXPERIMENTS AND RESULTS

This chapter discusses the evaluation techniques deployed to measure the performance of the proposed SLAM technique. These evaluation methods are popular measurement schemes employed by previous researchers because of their reliability. Using these evaluation techniques, the comparison of results obtained was used to reveal the SLAM technique with the best performance. Experimental validation was carried out to achieve the purpose below:

- Measuring the effectiveness of the proposed SLAM technique
- Estimating the robot trajectory in the environment.
- Estimating the computational cost as related to processing speed/time.
- Measuring the robot positioning in the environment.

In this study, the probabilistic modelling coupled with the Bayes rule presented in equation 4-25 was used to generate the simulated map and the robot trajectory for the proposed SLAM technique (DIK-SLAM) was simulated by an image processing software known as Matlab. Matlab is a simulation software that supports multi-paradigm numerical computing environment and propriety programming language, developed by MathWorks (Teng, 2000). The software was employed to carry out this test because it allows the implementation of an algorithm, plotting of data/function and matrix manipulation, which is a major requirement for achieving the objectives of this study. Thus, this is not only limited to these three functions, it also supports the creation of user interface and accommodates codes within other programming languages like C/C++. However, the experiment is not only limited to simulation software, data collected are equally important because they are measured, reported and analysed to visualize the performance of the technique proposed using graphs, images, etc. In this study, public and private datasets are used for validation of various SLAM techniques and will be discussed in this chapter. These datasets are all evaluated based on Absolute Trajectory Error (Absolute pose difference). Given that a robot is navigating in an environment, the quantitative measurement was carried out by using Absolute Trajectory Error (ATE) at time step k for every distance covered and was compared to ground truth estimate for error measurement. Illustration of ATE is expressed below in equation 5-1.

$$\mathrm{F}_{\mathrm{k}} = SP_k\left(Q_k^{-1}\right) \qquad 5\text{-}1$$

The F_{k} represents the ATE at a time step k, which can be calculated by comparing the absolute distance between ground truth and simulated trajectory. As these trajectories are specified in an arbitrary coordinate frame, they need to be aligned first. This is accomplished using the method proposed in (Shalal et al., 2015., Kaser, 2019). Furthermore, the F_{k} finds the euclidean transformation S (geometric transformation of euclidean space that preserves the euclidean distance between every pair at a point) corresponding to the least-square solution that maps the estimated trajectory $P_{1:m}$ onto the ground truth trajectory $Q_{1:m}$ (Kaser, 2019).

5.1 Experiment 1: Evaluation with Publicly Available Datasets

In this section, the publicly available dataset was employed for evaluation and the presented DIK-SLAM technique was evaluated using a dataset known as TUM RGBD. The TUM RGBD is originally owned by the computer vision group at TUM. This dataset was generated from a Microsoft Kinect RGB-D camera sensor highly accurate and time-synchronized ground-truth poses from a motion-captured system. This sensor can provide a sequence of colour and depth images of 640*480 resolution at 30Hz video frame rate. The TUM RGB-D is a huge dataset equipped with a sequence of RGB-D data and ground truth trajectory estimate for a different sequence of data. This dataset consists of numerous sequences of environments but for this study only four of these sequence were employed and are discussed in Table 5-1 (Kaser, 2019).

Table 5-1: TUM RGB-D dataset description

Environment Sequences	Description
Freiburg2_desk_ with_person	This environment is an office scene with a person moving and interacting with some object in the environment. This scene was selected in this study to check the SLAM algorithms robustness in a dynamic situation. The scene characteristics are provided below, which makes validation with other SLAM algorithms achievable. Duration: 142.08s Duration with ground-truth: 119.37s Ground-truth trajectory length: 17.044m Avg. translational velocity: 0.121m/s Avg. angular velocity: 5.340deg/s Trajectory dim.: 2.30m x 3.93m x 0.51m. File size: 2.71GB. Thus, this scene information was last updated on 30 September 2011 at 15:17 pm.
Freiburg2_360_ kidnap	In this environment scene, the sensor was covered numerous periods while pointing to a different location to generate an environment that the robot will be unable to recognise, which will make pose tracking difficult. Since the study targeted kidnapping and loop closure, this scene was selected to see if SLAM algorithms could identify and

	recover from kidnapping and loop closure. The scene characteristics are provided below: Duration: 48.04s Duration with ground-truth: 47.49s Ground-truth trajectory length: 14.286m Avg. translational velocity: 0.304m/s Avg. angular velocity: 13.425deg/s Trajectory dim.: 4.26m x 3.44m x 0.12m File size: 0.89GB. Thus, this scene information was last updated on 30 September 2011 at 15:15 pm.
Freiburg3_structure _texture_far	This scene was created by moving the sensor through an environment with much texture and structure,since the study targeted environmental noises. The scene was selected because texture and structure in the image cast different forms of illumination variation that allows testing the SLAM algorithms ability to overcome environmental noises. The scene characteristics are provided below: Duration: 31.55s Duration with ground-truth: 31.56s Ground-truth trajectory length: 5.884m Avg. translational velocity: 0.193m/s

	Avg. angular velocity: 4.323deg/s Trajectory dim.: 1.90m x 4.56m x 0.08m File size: 0.55GB. Thus, this scene information was last modified: on 07 August 2012 at 18:55 pm
Freiburg3_nostructure _notexture_near _withloop	This environment scene represents an industrial hall recorded intentionally with little visible structure and features. This scene was selected because this technique relies heavily on predefined objects and in the absence of these features, the performance is affected (Wurm et al., 2003). This experiment is crucial because the SLAM algorithms proposed for evaluation are particle-based and will like to monitor their behaviour under low numbers of feature and texture present in the environment with their ability to identify previously visited position. Duration: 37.74s Duration with ground-truth: 37.72s Ground-truth trajectory length: 11.739m Avg. translational velocity: 0.319m/s Avg. angular velocity: 11.241deg/s Trajectory dim.: 2.98m x 5m x 0.34m File size: 0.45GB. Thus, this scene information was last modified: on 07 August 2012 at 18:57pm.

These sequences of environment datasets are used to analyse and test the robustness of various SLAM algorithms and can be found on this site: https://vision.in.tum.de/data/datasets/rgbd-dataset. The TUM RGB-D is a popularly used dataset most especially in visual SLAM, since the study proposed a visual SLAM, this dataset was employed to test for comparison with the Monte-Carlo algorithm. In this section, the simulation was carried out using Matlab for a comparison between DIK-SLAM and the Monte-Carlo algorithm. The qualitative trajectory results were validated by using the Root Mean Squared Error (RMSE) overall time indices of the translational component as related to the Absolute Trajectory Error (ATE). An expression is provided below in equation 5-2.

$$RMSE(F_{1:m}) := \left(\frac{1}{m} \sum_{k=1}^{m} \left\| trans(F_k) \right\|^2 \right)^{\frac{1}{2}} \qquad \text{5-2}$$

where $trans\ (F_k)$ represents the translation component of the Absolute Trajectory Error F_k for every time step k and m is the total number of time step in the sequence. Thus, the result obtained from the Root Mean Squared Error (RMSE) evaluation is presented in Figure 5-2. This assessment technique was employed because of easy comparison in case we are interested in comparing our result with other researchers study since it is common technique of evaluation but for now our attention of comparison is focused on the Monte-Carlo algorithm

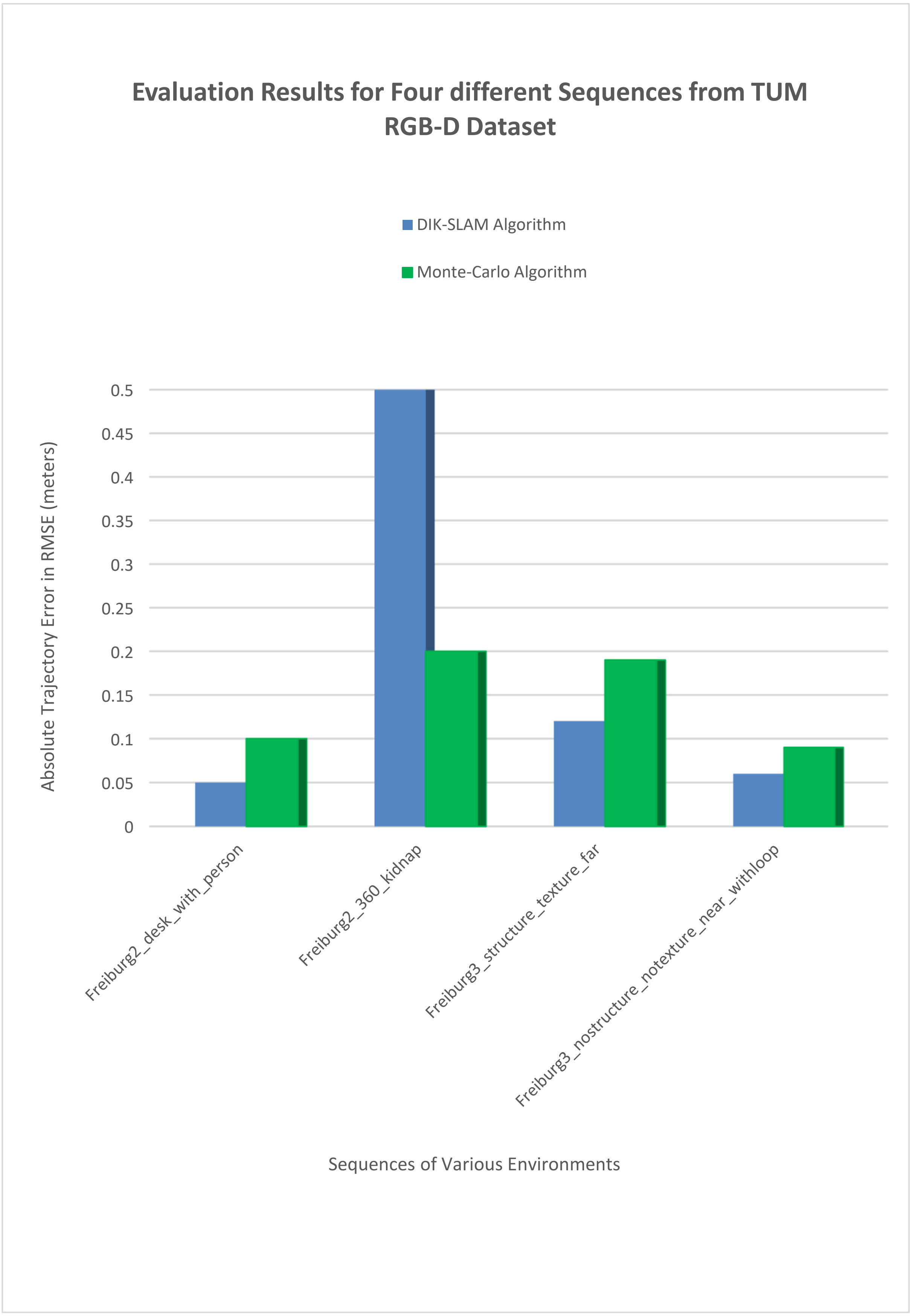

Figure 5-1: Absolute trajectory error measured using RMSE results of TUM RGB-D dataset

Given the comparison experiment presented between DIK-SLAM and Monte-Carlo algorithms in Figure 5-1, the results showed that DIK-SLAM represented with a blue bar of the chart performed better than the Monte-Carlo algorithm represented in the green bar of the chart. The result for every sequence of image presented in Figure 5.1 showed the DIK-SLAM attaining lower RMSE of ATE in its trajectory, compared to the Monte-Carlo algorithm. Moreover, this was the study expectation since the original Monte-Carlo algorithm had been modified to present the DIK-SLAM. However, in the Freiburg2_360_kidnap experiment, the original Monte-Carlo algorithm attained a lower RMSE of ATE in its trajectory compared to DIK-SLAM. Thus, investigation after comparison with the ground truth discovered that the Monte-Carlo algorithm got kidnapped and could not recover. Therefore, RMSE of ATE in its trajectory was only estimated to the point where it got kidnapped. But in the case of DIK-SLAM, it was able to cover more distance because of its ability to recover from kidnapping. This accomplishment in DIK-SLAM further allows the continuity of measuring RMSE of ATE in its trajectory, beyond where the Monte-Carlo algorithm experienced kidnapping. Moreover, this situation allows DIK-SLAM to attain higher RMSE of ATE in its trajectory than in the Monte-Carlo algorithm. Nonetheless, the margin of the higher RMSE of ATE in its trajectory of the Freiburg2_360_kidnap experiment compared to other environment scenes is the highest because of the frequent occurrence of kidnapping. Thereafter overcoming kidnapping, a slight significance error occurs in its pose after recovering and its accumulation contributes to why DIK-SLAM attained the highest RMSE of ATE in its trajectory of the Freiburg2_360_kidnap experiment compared to other environment scenes. Meanwhile, comparing this study with the work of other researchers that employed a similar dataset, the result obtained is in accordance (Kaser, 2019).

5.2 Experiment 1.1: Evaluation with Publicly Available Dataset based on Relative Pose Error Translational using RMSE

The environment sequences presented in Table 5.1 were further evaluated for pose error using Relative Pose Error (RPE). The RPE is an important measurement that must be carried because it corresponds to the drift of the robot trajectory from the ground truth measurement. This becomes useful for evaluating the visual odometry system. Thus, the relative pose error metric allows rotational and translational error to be combined to a single measure, but the rotational errors are indirectly captured by the Absolute Trajectory Error (ATE). The two measurements are strongly connected; therefore, the study's relative pose error measurement will only be limited to translational error. The relative pose error translational (RMSE) metric used for evaluating the global error of trajectory at the time step k is presented in equation 5-3.

$$RMSE(E_{1:m},\nabla)=\left(\frac{1}{n}\sum_{k=1}^{n}\left\|trans(E_k)\right\|^2\right)^{\frac{1}{2}} \qquad 5\text{-}3$$

where

$n=m-\nabla$,

$trans(E_k)$ signifies the translation components of the relative pose error E_k, m represents a sequence of camera poses and n represents individual relative pose errors along the sequence, while ∇ represents a fixed time interval, which is an intuitive value. In this study, the fixed time value is 1 and $RMSE(E_{1:m})$ then gives the drift per frame as presented in equation 5-4

$$RMSE(E_{1:m})=\frac{1}{m}\sum_{\nabla=1}^{m}RMSE(E_{1:m},\nabla) \qquad 5\text{-}4$$

Equation 5-4 was used to extract the relative error pose for the sequence of the environment on an average overall of time intervals. Illustration of the result is presented in Figure 5-2.

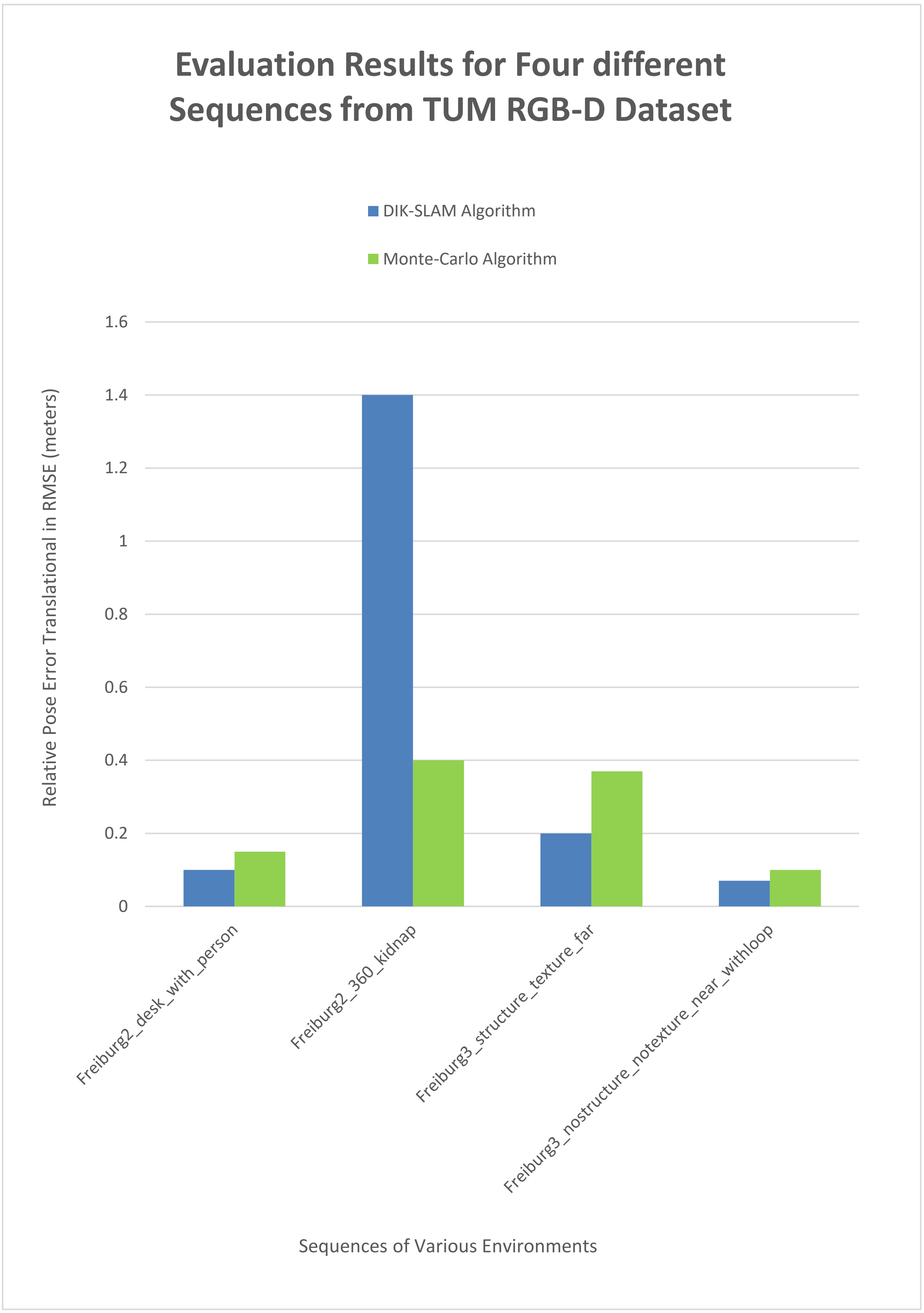

Figure 5-2: Relative pose error Translational measured using RMSE results of TUM RGB-D dataset

Figure 5-2 presents a comparitive result between DIK-SLAM and Monte-Carlo algorithms, which is similar to the one obtained in Figure 5-1. The results displayed, support DIK-SLAM signified by a blue bar of the chart attaining better performance than the Monte-Carlo algorithm presented in the green bar of the chart for every sequence of environments. The DIK-SLAM was declared to have better performance because it attained lower relative pose translation error (RMSE) in its trajectory, compared to the Monte-Carlo algorithm. This was expected since the Monte-Carlo algorithm had been modified to present the DIK-SLAM. Hence, in the Freiburg2_360_kidnap experiment, the DIK-SLAM attained higher relative pose translation error (RMSE) in its trajectory than the Monte-Carlo algorithm. Thus, investigaton into the ground truth discovered that the Monte-Carlo algorithm could only cover limited trajectory because it got kidnapped and could not recover. Therefore, a limited measurement was only allowed on its trajectory up until where it experienced kidnapping, unlike the DIK-SLAM algorithm, which covers more trajectory because of its ability to recover from kidnapping. These recoveries lead to more accumulation of relative pose translation error (RMSE) in its trajectory, which contributes to the reason, DIK-SLAM attained higher value than the Monte-Carlo algorithm. Nevertheless, the margin of the translation (RMSE) error in the Freiburg2_360_kidnap experiment compared to other environment scenes is the highest because of the frequent occurrence of kidnapping. Thereafter recovery from kidnapping, more error occurs in its pose and its accumulation contributes to DIK-SLAM attaining the highest translation (RMSE) error in Freiburg2_360_kidnap, compared to other environment scenes. Thus comparing the study result with the publicly available results that employed a similar dataset, the result corresponds (Kaser, 2019).

5.3 Experiment 1.2: Evaluation with Publicly Available Datasets based on Processing Time

In this study, the SLAM algorithms is further evaluated by recording the time required to process an entire sequence of the environment, since the study is taking into consideration computational cost as related to processing speed. The result acquired is given in Figure 5-3.

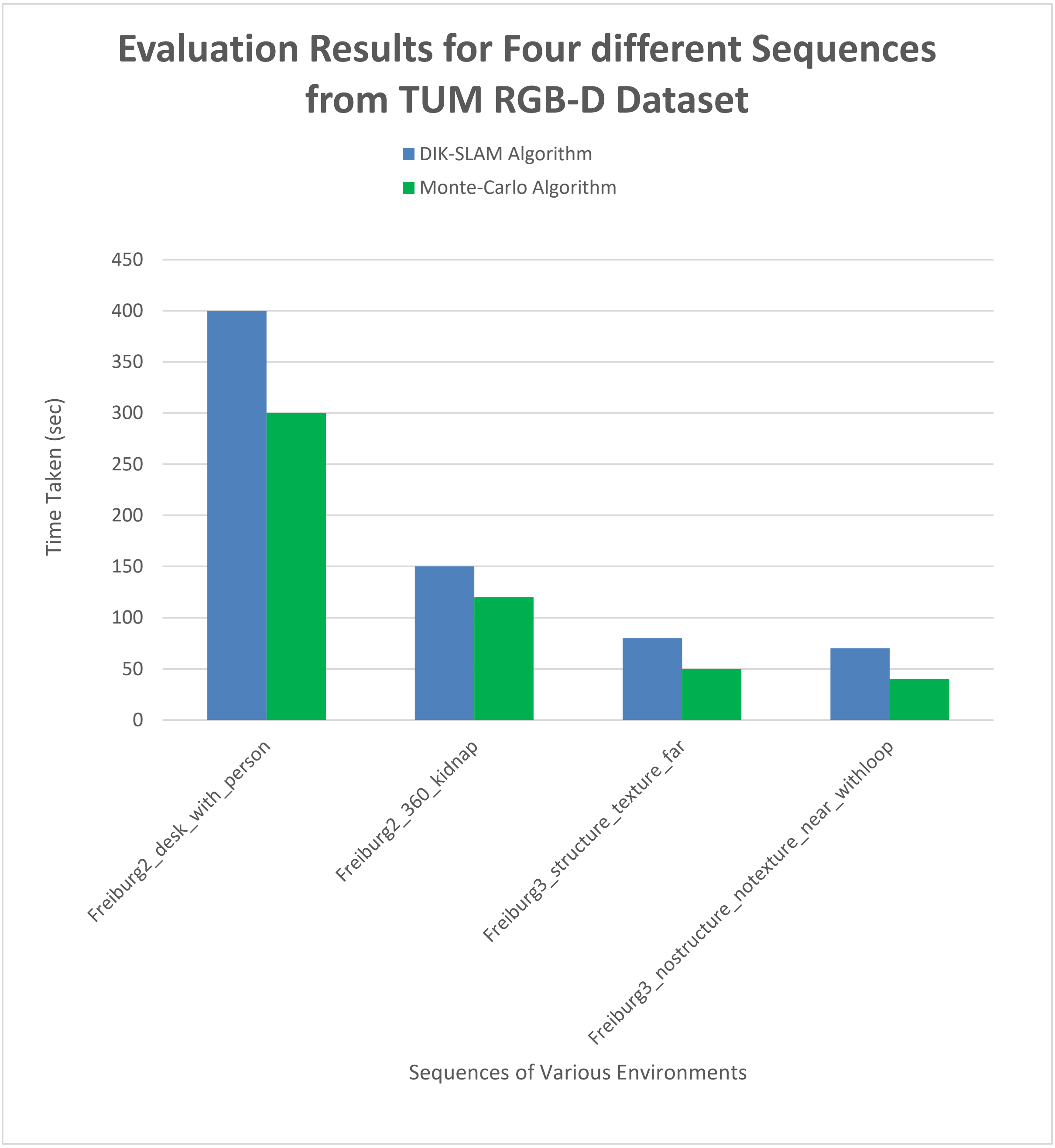

Figure 5-3: Time taken results to process sequences of Environment in TUM RGB-D datasets

Considering the comparison result between the DIK-SLAM and Monte-Carlo algorithm presented in Figure 5-3, the results displayed, support Monte-Carlo algorithm presented in the green bar of the chart attaining better performance than DIK-SLAM signified by a blue bar of the chart for every sequence of environments. This conclusion is because Monte-Carlo attained a lower processing time compared to the DIK-SLAM, which implies that processing speed in Monte-Carlo is faster than the DIK-SLAM algorithm (Wei et al., 2019). Investigation reveals that, the modification of Monte-Carlo to present DIK-SLAM with an improved performance has increased it computational complexity. The effect slowered the processing speed but considering the performance and the processing time margin between the two algorithms, the DIK-SLAM processing time is acceptable. Thus, the two algorithms attained the highest processing time at the scene Freiburg2_desk_with_ person, because the size of the file is huge compared to other environment scenes. The size of the data determines the processing time since the file is huge, it requires more processing time than other sequences of environments (Baeza-Yates and Liaghat, 2017).

5.4 Experiment 2: Evaluation with Private Dataset.

Given that the study presented a DIK-SLAM technique that was further investigated using a private dataset. This private dataset was recorded by a digital camera to generate multiple images with a resolution of 320 X 240 pixels. These images include features like the wall, floor, etc. representing samples used to guide the robot trajectory. The private dataset was generated from the electrical building of a university in South Africa, containing six (6) classrooms, three (3) offices with all their doors closed, one (1) hall used as cafeteria, two (2) corner ways that lead to two (2) staircases, two (2) corner ways that lead to the 2 lifts and passages. Hence, three SLAM algorithms were tested on the private dataset using Matlab simulation and were evaluated based on the ground truth measrement of the dataset. Thus, this ground truth was extracted using SICK-S 200, a 2D Laser scanner with a 270-degree field of view, 50-meter maximum range, 0.1 mini-meter resolution and a camera sensor based on the technique proposed by (Ming and Ji-Ying, 2010) and (Ceriani et al., 2009). These sensors has assisted to extract the ground truth measurement also the classification of a detailed map, illustration is presented in Figure 5-4a. The ground truth measurements were assumed to be without error, close to real life measurement and the safest path for any robot to follow. These ground truth measurements were used for comparison with the simulation results of the SLAM algorithms obtained from Matlab.

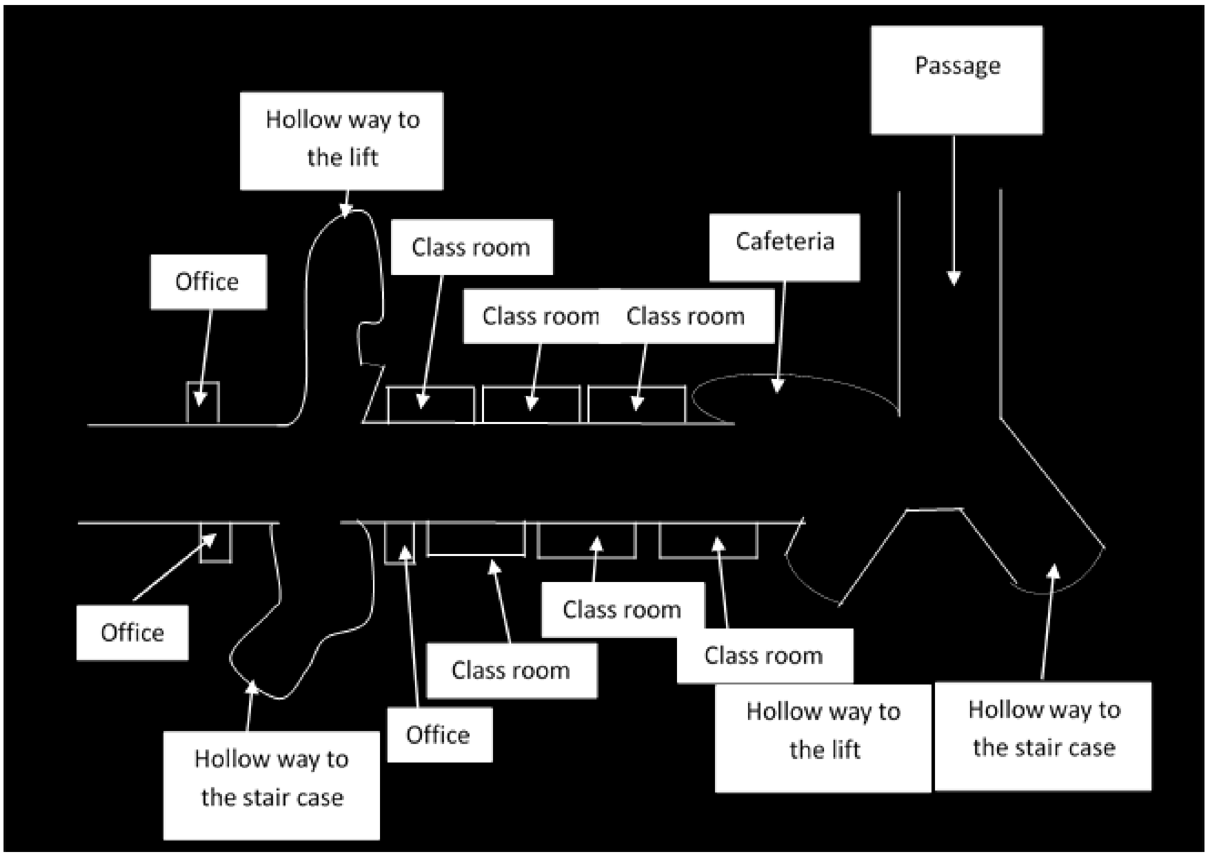

Figure 5-4a: The extracted ground truth of the generated dataset using laser sensors

The ground truth generated map presented in Figure 5-4a, represent an indoor environment of the private dataset and sample frame from the dataset is provided in Figure 5-4b.

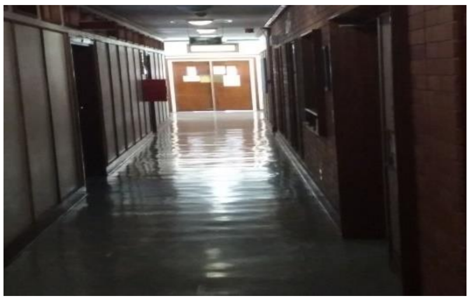

Figure 5-4b: Sample of generated dataset image used to test the SLAM algorithms

This dataset was intended for studying the performance of various SLAM algorithms and results obtained were discussed broadly in the experiment below. However, the private dataset was employed because of the access to the ground truth measurement of the environment which facilitate easy comparison with the experimental result obtained from the SLAM algorithms.

Experiment 2.1: Qualitative Experiment on Kidnapping Evaluation for DIK-SLAM VS Original Monte-Carlo Algorithm

Kidnapping in robotic is often experienced during navigation. This happens when the robot is not aware of its new position and given that there is no failing sensor, increasing in measurement noise is a common factor that can lead to it. The consequences lead to erroneous or inability to estimate pose, and in worst cases, can lead to kidnapping with no recovery, which violates the basic principle of SLAM. In the attempt to validate the proposed DIK-SLAM ability to overcome kidnapping, the private dataset was employed for evaluation. This dataset was first generated without the introduction of objects for monitoring the behaviours of the algorithms. Matlab simulation software was employed to demonstrate the map built and robot trajectory developed by the SLAM techniques. The robot trajectory is significant to show the robot pose at every distance covered in the map created and facilitate easy error measurement for these trajectories (Absolute pose difference when compared to ground truth) (Sturm et al., 2012). This experiment was implemented using Matlab simulation for a comparison between DIK-SLAM and the original Monte Carlo algorithm presented in the study of (Thrun et al., 2001). The original Monte-Carlo algorithm is publicly available and can be accessed from this site: https://ch.mathworks.com/matlabcentral/fileexchange/8068-montecarlo. Figure 5-5a shows the qualitative result of the SLAM algorithms.

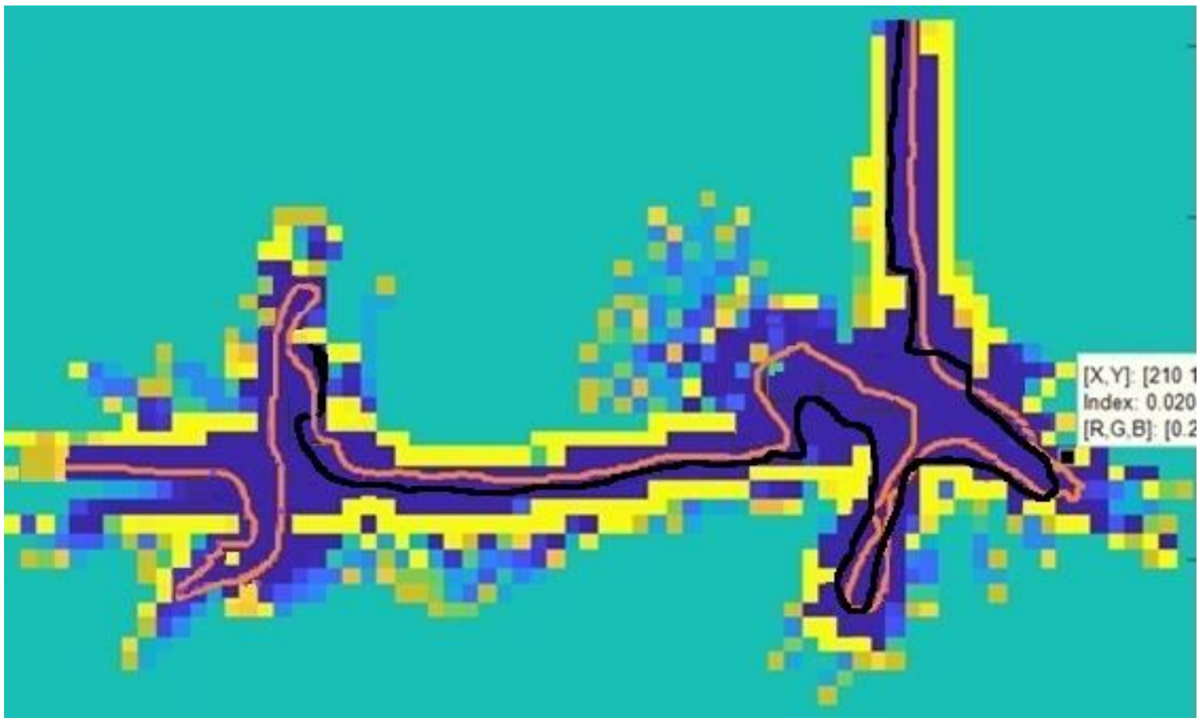

Figure 5-5a: Qualitative trajectory results for DIK-SLAM and Monte-Carlo algorithm

In Figure 5-5a, there are two different robot trajectories in the simulated map representing the environment. The orange colour represents the robot trajectory for DIK-SLAM, while the black colour represents the robot trajectory of the original Monte-Carlo algorithm. The simulated map is made up of two colours, the yellow part is boundaries/blockage or obstacles with a bit represented by one, while the blue part is road or drivable part with a bit represented by zero. This information is meant to assist in successful trajectory. At the beginning, the robot's trajectory was perfect but as they began to cover more distance, the black colour trajectory which signifies the Monte-Carlo algorithm often got to the yellow part (blockage) before correcting its step back to the blue part (Road/drivable) and this was experienced in many parts of the map in Figure 5-5a. This trajectory error will be measured quantitatively to demonstrate the SLAM algorithm with the best performance. The Absolute Trajectory Error (ATE) will be employed to measure the errors in the robot trajectories. Furthermore, it was discovered that the robot is kidnapped because the black trajectory could not get to the destination, as a result of not recovering from the kidnapping position, whereas the orange representing DIK-SLAM was able to reach its destination. Thus, the area where the kidnapping happened on the dataset is a complex environment, considering a corner suddenly ends by an obstruction (wall) and continues after the obstruction (wall). Furthermore, it could be the unforeseen noises in the natural form of the image that triggers kidnapping because when the image was matched to the simulated map, where the kidnapping occurred, heavy shadow and lighting noises were present. The presence of such noises adulterate the RGB colour properties. The effect might changed the appearance of the image, making it difficult for the robot to recognise it. Moreover, the study further justified this reason by performing another experiment of increasing the image noise to monitor the behaviour of these SLAM algorithms. Environmental noises, like any other noise, degrade image quality which causes loss of data and dissatisfactory in visual consequence, as a result of distortion in the RGB properties of the image (Alqadi, 2018). However, the study only targets shadow and light intensity and to increase such noises in the images, the attention is focused once again on the Matlab to increase the noise measurement in the image. Environmental noises alter the number of occurrences of colour mixture that ends up changing the appearance of the image, depending on the noise intensity (Alqadi, 2018). The effect makes interpretation in an image very difficult and could lead to miss-classification in images (Alqadi, 2018). Hence, introducing these noises into the image can occur naturally, but Matlab can be used to generate such noise in images. In Matlab, the ‘Imadjust’ function has been used by researchers to increase or decrease the level of shadow and light intensity in images based on the concept of contrast and brightness (Sinecen, 2016). The brightness of the image can be adjusted when adding or subtracting respectively a certain value to Gray Scale level of each pixel, while the contrast of the image can be adjusted by multiplying all pixels

$i,\ j$ by a certain value. Illustration for both brightness and contrast is presented in equations 5-5 and 5-6 respectively.

$$G(i,j) = F(i,\ j) + b\ \left(\begin{smallmatrix} b>0\, BrightnessIncrease \\ b<0\ BrightnessDecrease \end{smallmatrix}\right) \qquad 5\text{-}5$$

$$G(i,j) = F(i,\ j) * c\ \left(\begin{smallmatrix} c>0\, Contrast\, Increase \\ c<0\ Contrast\, Decrease \end{smallmatrix}\right) \qquad 5\text{-}6$$

where $G(i,j)$ is the pixel representation of the image in the Gray Scale level and $F(i,j)$ represents the pixel of the input image (Sinecen, 2016).

The ‘Imadjust’ function based on equation 5-5 and 5-6 is used to increase both the brightness and contrast. Thus, increasing the brightness will severely increase the light intensity noise already affecting the image (Shuang et al., 2014) while increasing the contract will intensify the shadow noise by making it darker (Jack et al., 1999). Considering this experiment, the percentage_alteration of the noise ratio in the private dataset that was generated from the electrical building of a university in South Africa was increased by 10%. The experiment between the DIK-SLAM and the original Monte-Carlo algorithm was implemented once again to study and compare the SLAM algorithms behavior towards the new dataset with increased environmental noises. Illustration of the result obtained is provided in Figure 5-5b

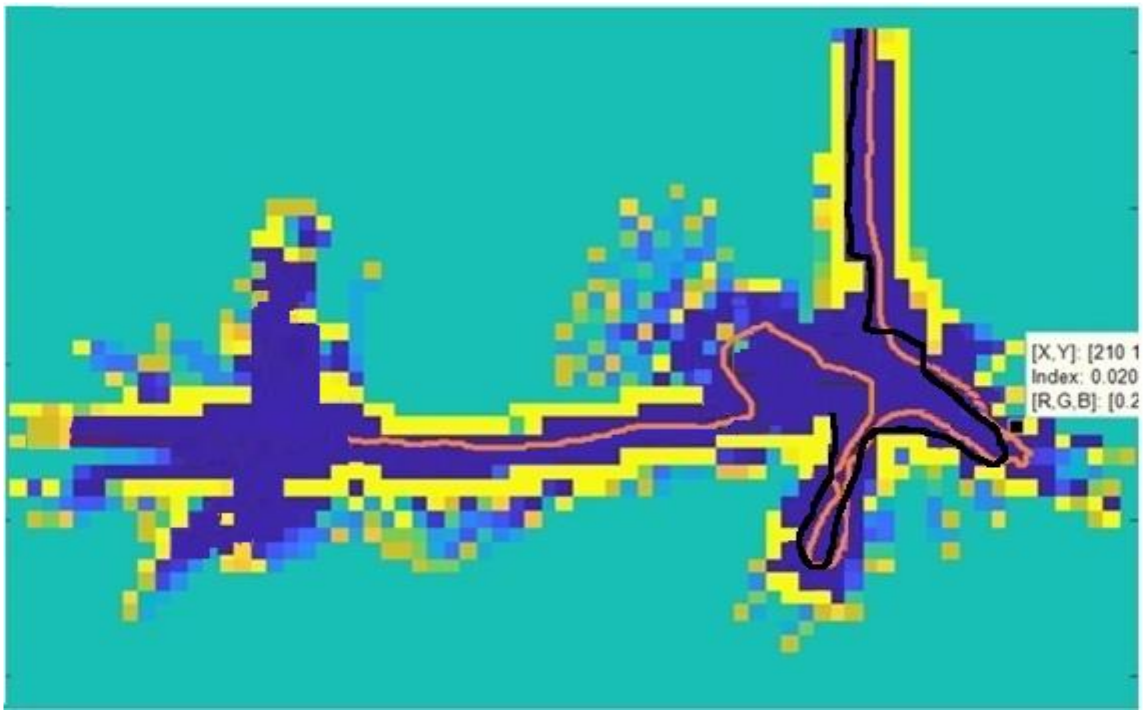

Figure 5-5b: Qualitative trajectory results for DIK-SLAM and Monte-Carlo algorithm in 10% increase noise level

In Figure 5-5b, the orange colour still signifies the DIK-SLAM trajectory, while the Black colour represents the trajectory of the Monte-Carlo algorithm. The simulated map is represented in two colours, the yellow part still symbolizes boundaries/blockage or obstacles with a bit denoted by one, while the blue symbolizes road or drivable part with a bit denoted by zero.

This information is meant to assist in a successful trajectory. In the beginning, the robot's trajectory is perfect but as they begun to cover more distance, the original Monte-Carlo algorithm got kidnaped earlier than in Figure 5-5a, just right after the second corner and could not recover.

Meanwhile, the DIK-SLAM proceeded to cover more distance even after the second corner in its trajectory, but eventually got kidnap and could not recover. Thus this experiment proved that DIK-SLAM could only withstand kidnapping in Figure 5-5a and to some extent, in 5-5b, because of the filters' ability to reduce the effect of the noises and its ability to re-localize itself. However, this study shows kidnapping does not only occur when there is a failing sensor, or when the robot is teleported into an unknown environment. Kidnapping can occur during severe/high intensity of noises (illumination variation and measurement noises) as supported by this experiment. This experiment also supports the effectiveness of the filtering algorithm employed towards addressing the problem of shadow and light intensity. Conclusively, Figures 5-5a and 5-5b that represent the qualitative results and for a comparison with both trajectories, support the DIK-SLAM ability to attain better performance than the original Monte-Carlo algorithm.

5.5 Experiment 2.2: Quantitative Result for Trajectory Error Measurement of the DIK-SLAM and Original Monte-Carlo Algorithm based Absolute Pose Difference.

This section discusses a different type of evaluation scheme used to validate the DIK-SLAM. This evaluation is carried out on the initial qualitative result of the private dataset generated from an electrical building of a university in South Africa without adjusting the brightness and contrast as presented in Figure 5-5a. The purpose of the experiment this time around, is to measure errors in the trajectories using the absolute pose difference as related to the ground truth robot trajectory for a comparison between DIK-SLAM and the original Monte-Carlo algorithm. There are several techniques that can be employed to implement error measurement, one of which is the pairwise error probability (Nobelen and Taylor, 1996, Nezampour et al., 2011). In this study, the Absolute Trajectory Error (absolute pose difference) presented in equation 5-1, will be employed for estimating the error in the robot trajectory as related to the ground truth trajectory. Illustration of the results obtained is given in Figure 5-6.

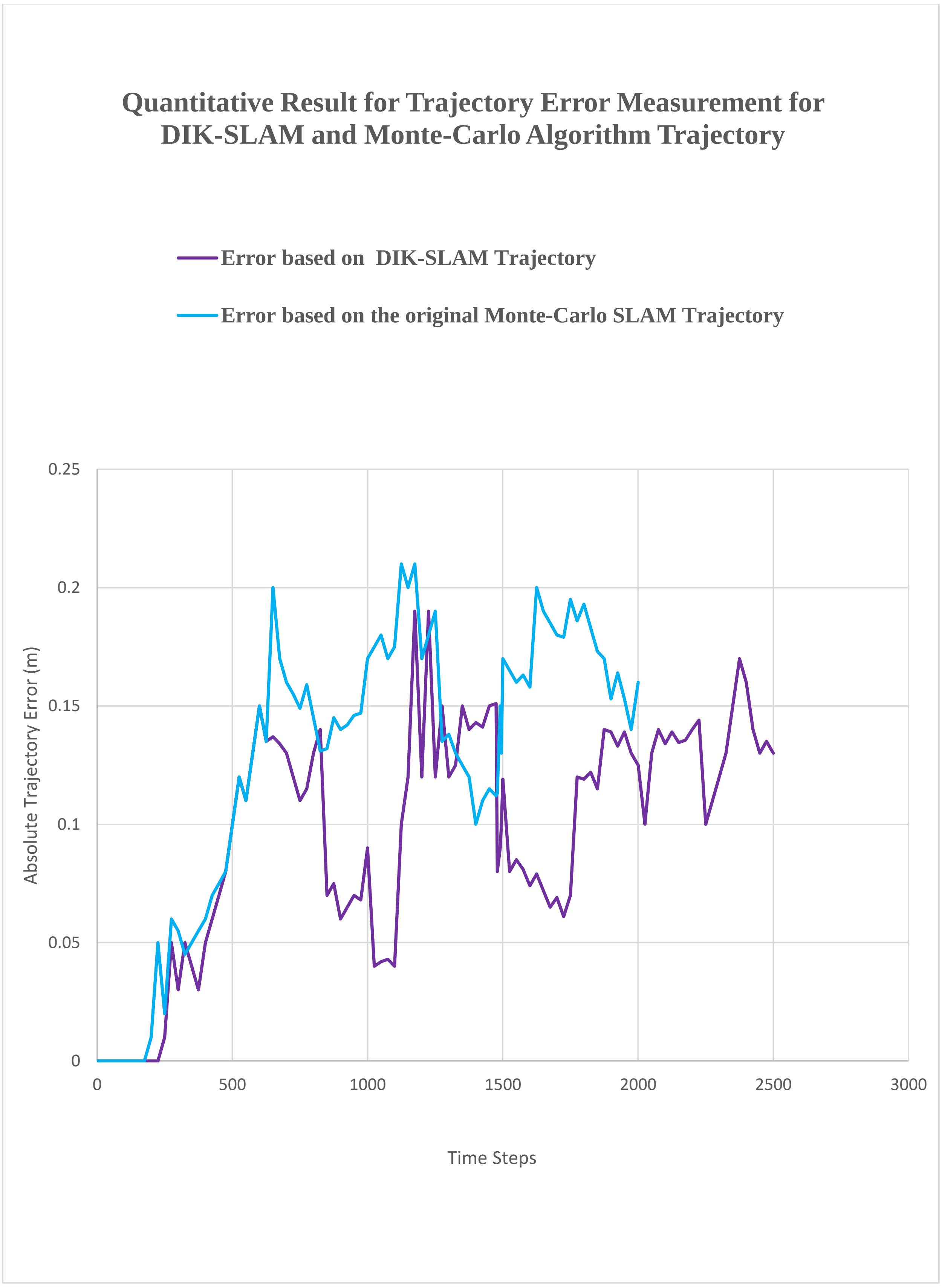

Figure 5-6: Quantitative result for trajectory error measurement for DIK-SLAM and Monte-Carlo algorithm

Figure 5-6 signifies the quantitative result of the error measurement derived from the absolute pose difference as related to the ground truth trajectory. The purple line colour in Figure 5-6 represents the result of the absolute trajectory error in the DIK-SLAM Trajectory, while the blue colour in Figure 5-6. represents the result of the absolute trajectory error in the original Monte-Carlo algorithm trajectory. Considering the observation in Figure 5-6, the absolute trajectory error between the two trajectories at the beginning is relatively low, but as much distance is covered into the areas where the images are suffering from severe lighting, shadow effect and corners, etc.

The absolute trajectory error between both algorithms becomes significant. The purple line colour which represents the DIK-SLAM trajectory attained a lower absolute trajectory error in most cases of the trajectory. The blue colour which signifies the original Monte-Carlo trajectory attained a higher absolute trajectory error for most trajectory covered. Its trajectory could not reach the final destination unlike DIK-SLAM because it could not recover from kidnapping. However, there are instances in the quantitative results, where both SLAM algorithms attained higher absolute trajectory errors in their trajectory. This is due to the presence of corners suddenly appearing at the edge of the passages. These places are challenging and it is where misdetection mostly appears in the qualitative results.

Given the relationship between accuracy and error, accuracy means the closeness of agreement between a true value and the measured value while error signifies differences between a measurement and the true value (Kitchenham et al., 2001). Thus, the higher the error, the lower the accuracy in trajectory and vice versa (Kitchenham et al., 2001). Based on the experimental results presented in Figure 5-6, the DIK-SLAM with lower error, attained a higher accuracy trajectory than the original Monte-Carlo algorithm. Overall, the performance of the DIK-SLAM algorithm is better than the original Monte-Carlo algorithm.

5.6 Experiment 2.3: Qualitative Experiment for DIK-SLAM Dynamic Environment Evaluation

In literature, SLAM algorithms were first introduced to address static environments (Thrun et al., 2001), but not all environments are static, there are environments that are dynamic in nature due to the presence of moving objects. Therefore, we observe the performance of the DIK-SLAM in a dynamic environment. In the DIK-SLAM technique, cells in the maps are altered as changes happen in the environment, since the concept relies on the assumption that cells in the map are independent of an object. Therefore, identifying changed cells in the map requires a constant comparison of the previous map with the current one to reveal the status of the cell that has changed and identified as a dynamic environment. In this experiment, the initial private dataset generated from an electrical building of a university in South Africa, was created without the object and was converted to a dynamic environment by first introducing 35 objects into the environment. 5 objects are introduced to the short passage, 14 objects to the long object and 8 objects to the hall. While 8 objects at an average of 2 are introduced at every corner (4) into the environment. These objects include boxes, bins, tables, chairs and people, all stationed in the environment before recording, using a digital camera to generate a dataset with objects. Illustration of the result for the object introduced in the environment is presented in Figure 5-7.

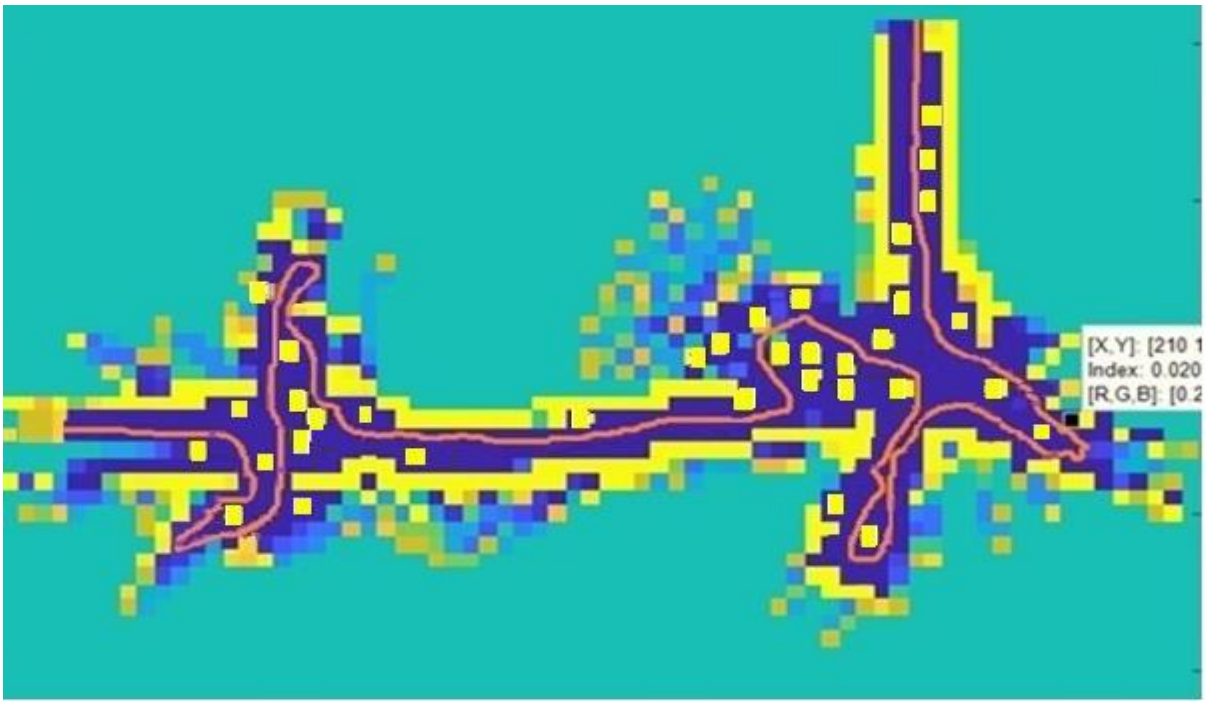

Figure 5-7: Qualitative result of DIK-SLAM on object detection

In Figure 5-7, there is a robot trajectory in orange colour representing DIK-SLAM in the simulated map representing the environment. The simulated map is made up of two colours, the yellow part is boundaries/non-drivable/blockage or obstacles with a bit represented by one, while the blue part is road/free/drivable part with a bit represented by zero just like Figure 5-5a. Thus, the difference between the mapping of Figure 5-7 and Figure 5-5a, which is an environment without an object, is that 35 certain cells of the map labeled blue in Figure 5-5a have changed to yellow in Figure 5-7. These cells are occupied by an obstacle/object, therefore, their status changed to yellow appearing in the mist of blue as presented in Figure 5-7. After the introduction of the object into the environment, we proceeded to the dynamic experiment by completing the removal of some objects/obstacles out of the environment. 3 objects were removed completely from the short passage, 10 objects were remove completely from the long passage, 4 objects were removed completely from the hall while 1 object was removed completely from every corner (4). Afterwards, the proposed concept that relies on the previous status to predict the current state is applied to compare the previous map with the current one to identify a cell that has changed its colour status and will be identified by x representing part of the environments that are dynamic in nature. The illustration is given in Figure 5-8a.

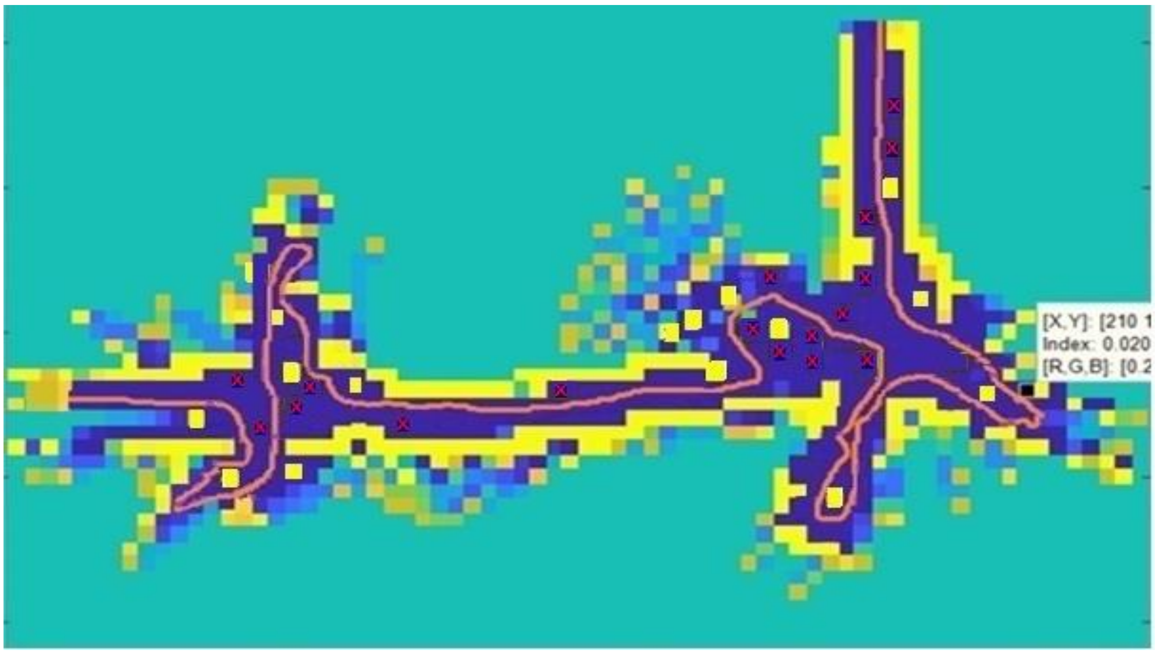

Figure 5-8a: Qualitative result of DIK-SLAM on the dynamic environment

In Figure 5-8a, the orange colour trajectory represents the DIK-SLAM navigation within a simulated map that is developed with two colours. The yellow part is boundaries/non-drivable/blockage or obstacles with a bit represented by one, while the blue part is road/free/drivable part with a bit represented by zero just like Figure 5-7. Thus, the difference between the mapping of Figures 5-7 and 5-8a is the DIK-SLAM ability to identify yellow cells (occupied by object) that are changed to blue (object free) and mark x representing the dynamic environment where, an object has been removed completely. While the yellow cells (occupied by object) in the mist of blue signifies the object that was not removed from the environment. The result presented in Figure 5-8a validated the DIK-SLAM ability to detect and cope with a dynamic environment. Although not all dynamic environments initiated were detected but the evaluation was carried out to determine the cause and the success rate of dynamic detection. These will be presented at the quantitative experiment. Furthermore, since the study targets dynamic noisy environments, it is important to study the behavior of the proposed SLAM algorithm in various noise level ratios. Thus, using the same dataset used to present the dynamic result in Figure 5-8a, the private dataset generated from an electrical building of a university in South Africa is already affected by shadow and light intensity noise. The study further intensifies these noises using the concept of brightness and contrast based on the "Imadjust" function of the Matlab as presented in equation 5-5 and 5-6. The brightness was increased to intensify the light intensity while the increase of contrast will intensify the shadow by making it darker. They both increased at a ratio of 5% before re-running the simulation once again, and an illustration of the result obtained is presented in Figure 5-8b.

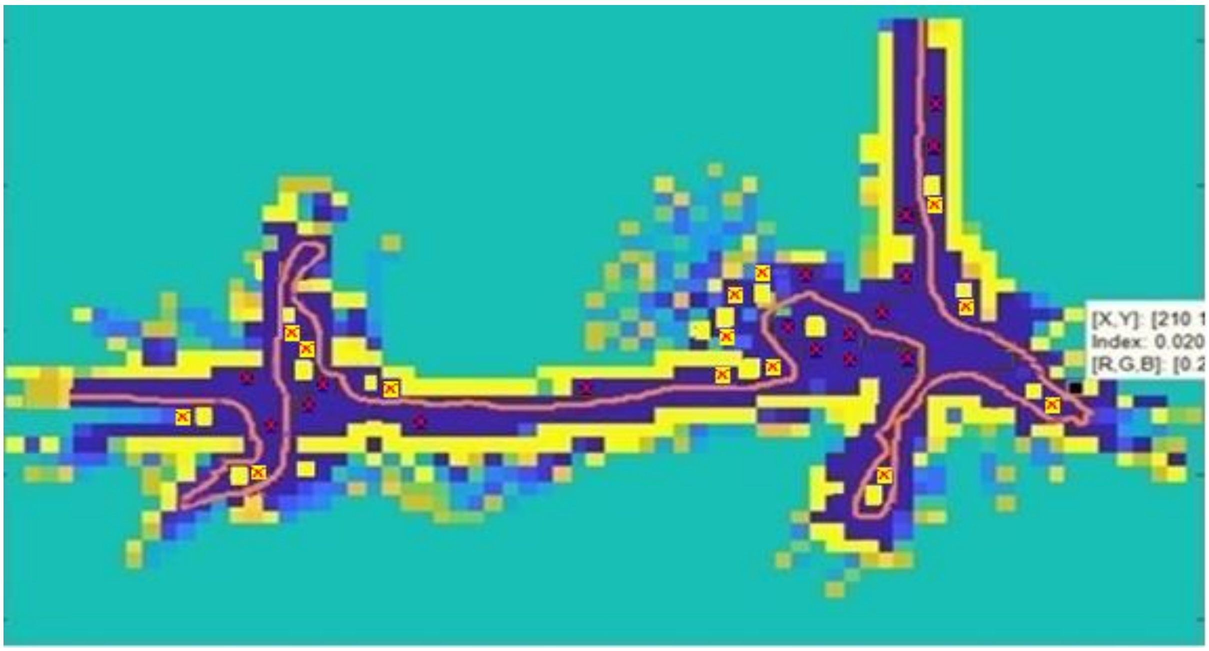

Figure 5-8b: Qualitative dynamic result for DIK-SLAM algorithm in 5% increase noise level

In Figure 5-8b, the orange colour trajectory represents the DIK-SLAM navigation within a simulated map that is developed with two colours. The yellow part is boundaries/non-drivable/blockage or obstacles with a bit represented by one, while the blue part is road/free/drivable part with a bit represented by zero. Thus, the DIK-SLAM was able to identify yellow cells (occupied by object) that are changed to blue (object free) and mark x similar to Figure 5-8a representing the dynamic environment, where the object has been removed completely. While the yellow cells (occupied by object) in the mist of blue signifies the object that was not removed from the environment. However, in Figures 5-8a and 5-8b, they both were unable to detect all true positive dynamic environments initiated. Thus, the difference between them was that there were false positive detections of the dynamic environment in Figure5-8b. In the simulated map, certain cells labelled blue, changed to yellow and was mark x since the cell status changed even though there was no object placed in that surrounding to initiate a dynamic environment. This will be evaluated quantitatively to determine the cause and the rate of dynamic false positive detection attained ,and the outcome will be discussed in experiment 2.4.

5.7 Experiment 2.4: Quantitative Experiment for DIK-SLAM Dynamic Environment Evaluation

Given that DIK-SLAM has shown the ability to cope with the dynamic environment, the study has not measured the rate of success attained by the SLAM algorithm. In carrying out this experiment, the qualitative result of the dynamic dataset generated from an electrical building of a university in South Africa as presented in Figure 5-8a will be evaluated. Thus, this qualitative result compared to the real-life scene will be evaluated to measure the success rate of the DIK-SLAM algorithm. In the real-life dataset, 35 objects were originally introduced into the environment, a total of 21 objects were removed from their initial position to initiate dynamic condition. However, in the simulation result presented in Figure 5-8a, not all the dynamic environments present in the dataset were detected. Therefore, it is important to measure the DIK-SLAM performance and the result obtained is presented below in Table 5-2.

Table 5-2: Quantitative result on DIK-SLAM dynamic environment evaluation

Environment Section	**Real-Life Object introduced to the environment**	**Simulated object detected in the environment**	**Real-life object removed to initiate dynamic environment**	**Simulated dynamic environment detected**
Total objects in the short passage	5	5	3	3
Total objects in the long passage	14	14	10	10
Total objects in the hall	8	8	4	4
Total objects at all the Corners	8	8	4	0
Total objects in the environment	**35**	**35**	**21**	**17**

Figure 5-8a has shown DIK-SLAM ability to cope with the dynamic environment, but did not display the success rate. Nevertheless, Table 5-2 based on the evaluation of the qualitative result presented in Figure 5-8a has helped to reveal the success rate of DIK-SLAM. In summary, every object removed to initiate a dynamic environment at both passages and the hall is perfectly detected. However, at the corners, 4 objects at an average of 1 for each corner was removed to initiate dynamic condition, but the DIK-SLAM could not identify any of the dynamic environments at the corners. This miss-detection happens because some of the objects are small and positioned very close to the wall, while the other miss-detection was caused by reflections from the walls.

The DIK-SLAM performance is satisfactory because of its ability to identify a total number of 17 dynamic environments compared to the real-life, where a total number of 21 dynamic environments were initiated. Overall, the percentage of dynamic detection for DIK-SLAM is estimated at 81% using equation 5-7.

$$dynamic\ performance = \frac{x_d}{x_t} \times 100 \qquad 5\text{-}7$$

where x_d represents the number of dynamic environments detected and x_t represents the total number of all dynamic environments initiated in the environment.

However, the impressive results of 81% of dynamic environment detected in the original dataset without any form of adulteration were considered.. The study, on the other hand, has not measured the success rate attained by the DIK-SLAM when the noises in the image are increased by 5%. In carrying out this experiment, the qualitative result of the dynamic dataset generated from an electrical building of a university in South Africa as presented in Figure 5-8b will be evaluated. Thus, this qualitative result was compared to the real-life scene to measure the success rate of the DIK-SLAM algorithm when encountering dataset with increased noise level. Illustration of the results obtained is presented in Table 5-3.

Table 5-3: Quantitative result on DIK-SLAM dynamic environment evaluation at 5% noise increment

Environment Section	**Real-Life Object introduced to the environment**	**Simulated object detected in the environment**	**Real-life object removed to initiate dynamic environment**	**Simulated dynamic environment detected**	**False-positive Simulated dynamic environment detected**
Total objects in the short passage	5	5	3	5	2
Total objects in the long passage	14	14	10	15	5
Total objects in the hall	8	8	4	7	3
Total objects at all the Corners	8	8	4	4	4
Total objects in the environment	**35**	**35**	**21**	**31**	**14**

Considering the results presented in Figure 5-8b, this once again supports the DIK-SLAM mapping technique's ability to cope with the dynamic environment, since the cell in the grid can change status. Nonetheless, it did not display the success rate but Table 5-3 based on the evaluation of the qualitative result presented in Figure 5-8b has helped to reveal the success rate of DIK-SLAM. Thus, a similar object removed to initiate the dynamic environment in Figure 5-8a was also used but under the influence of noise increment at a ratio of 5% to present the DIK-SLAM result in Figure 5-8b. In summary, at the short passage where 3 real-life objects were removed to initiate a dynamic environment, a total of 5 dynamic environments was detected but 2 of them were false positive because no objects were placed in that surrounding to initiate a dynamic environment. At the long passage where 10 real-life objects were removed to initiate a dynamic environment, a total of 15 dynamic environments was detected but 5 of them were false positive because no objects were placed in that surrounding to initiate a dynamic environment. At the hall/cafeteria where 4 real-life objects were removed to initiate a dynamic environment, a total of 7 dynamic environments was detected but 3 of them were false positive because no objects were placed in that surrounding to initiate a dynamic environment. While at the corners, a total of 4 real-life objects at an average of 1 per corner were removed to initiate a dynamic environment. Thus, a total of 4 dynamic environments were detected but all were false positive because no objects were placed in that surrounding to initiate a dynamic environment. However, investigating the cause for the false-positive detection led us to realign the pixel (i, j) of the dataset with increased noises to the map obtained in Fgure 5-8b. Thereafter, it was realized that point with false-positive detection is affected with high light reflection and shadow cast from objects. This discovery shows that the high intensity of light reflection and shadow cast from the object has the ability to trigger obstacle detection on the cells of the grip map. Therefore, cell status changed from blue to yellow, and it was marked x by the DIK-SLAM algorithm, since a dynamic condition has been initiated,although there is no real object placed in that surrounding to initiate such a condition.

5.8 Experiment 2.5: Qualitative Trajectory Experiment for DIK-SLAM VS EKF Algorithm in a Dynamic Environment.

The study further validates the performance of DIK-SLAM using the same dataset with the introduction of the dynamic environment as presented in experiment 2.3 for evaluation purposes. In this experiment, the Monte-Carlo algorithm was not selected because it can only cope with a static environment. Alternatively, an Extended Kalman filter proposed by another researcher was employed for this evaluation. Thus, the work of (Bailey et al., 2014) was selected because they discussed a new version of an Extended Kalman Filtering algorithm that could perform better than the original Extended Kalman Filter and can be publicly accessed from this site: www.acfr.usyd.edu.au/homepages/academic/tbailey/software/index.html. The availability of this algorithm has facilitated an easy comparison with the proposed SLAM technique (DIK-SLAM) presented in this study. The evaluation is based on the robot trajectory because of the interest in pose difference at every instance, taking into consideration the ground truth classification. Figure 5-9 represents the qualitative results obtained between DIK-SLAM and the modified Extended Kalman Filter (Bailey et al., 2014).

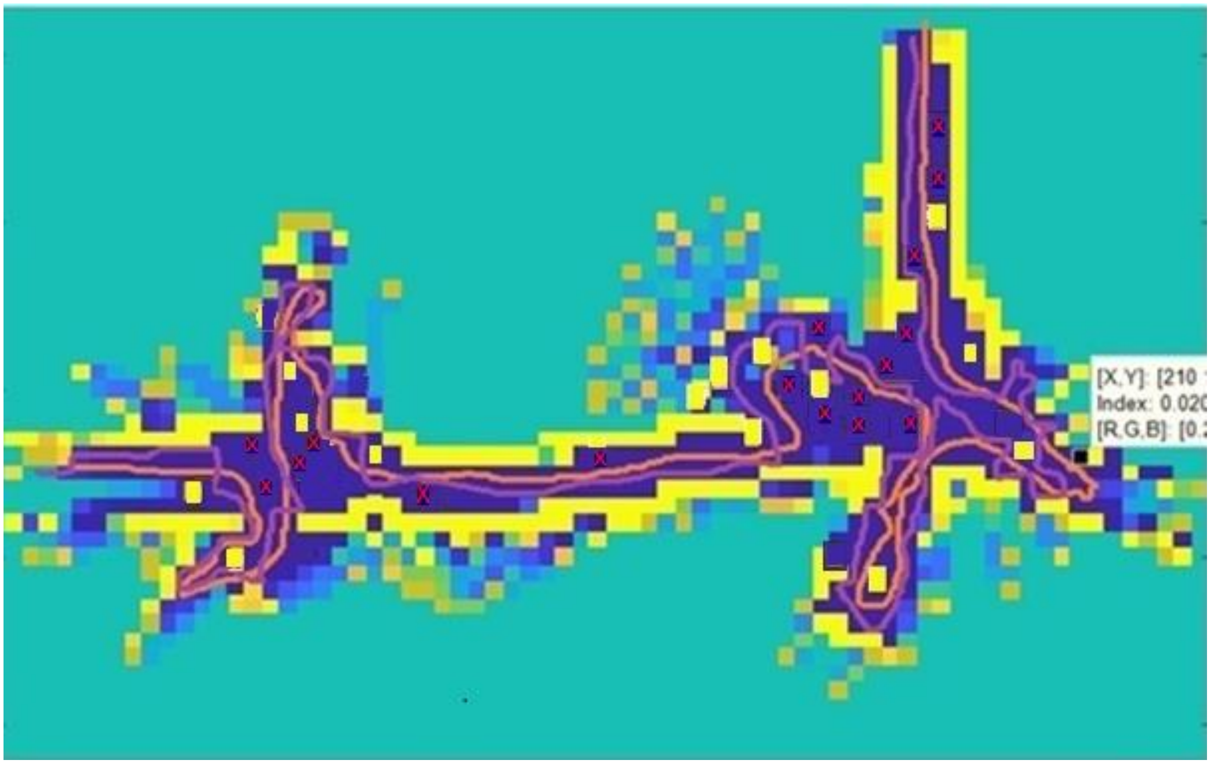

Figure 5-9: Qualitative trajectory results for DIK-SLAM and modified EKF algorithm

In Figure 5-9, the yellow part is boundaries/non-drivable/blockage or obstacles with a bit represented by one, while the blue part is road/free/drivable part with a bit represented by zero just like Figure 5-7. However, there are two robot trajectories navigating in the simulated model representing the environment, the DIK-SLAM trajectory still maintains the orange colour, while the robot trajectory in purple colour represents the modified EKF presented by (Bailey et al., 2014). In the simulated model, the yellow part that appears in the mist of blue region is classified by the system to be obstacle/object, because their bits are represented by one, since the cell of the map is occupied while those marked x are the detected dynamic environments. Given the trajectories of the robots, the movement of the two are very close, they are both able to cope with the dynamic environment, but there are instances that the purple trajectory which represents the modified EKF, get to the yellow part, which represents non-road/obstacle before correcting its trajectory to the blue, which signifies the drivable part. Thus, this similar incidence also happens with the orange trajectory, which represents DIK-SLAM, but in few instances compared to the modified EKF. This will be more justified during the quantitative experiment because we will be able to visualize better by means of a graph to display the rate at which the systems perform in their pose as related to their trajectory for every time step. However, in the qualitative results for Figure 5-9, the two algorithms performed closely in their trajectory, most importantly they could both cope with the dynamic environment and there was no instance of kidnapping without recovery. But, considering the miss-detection that frequently happens at the corners, the DIK-SLAM attained better performance than the modified EKF.

5.9 Experiment 2.6: Quantitative Result for Error Measurement of the robot Trajectory for DIK-SLAM and Modified EKF

This section discusses a different type of evaluation scheme used to validate the DIK-SLAM. The evaluation again is carried out on the qualitative result of the dynamic dataset generated from an electrical building of a university in South Africa as presented in Figure 5-9. The purpose of the experiment is to measure the error as related to the robot trajectory, taking into consideration the ground truth trajectory for a comparison between DIK-SLAM and modified EKF algorithm. There are several techniques that can be employed to carry out this task, one of which is the pairwise error probability. However, in this study, the Absolute Trajectory Error (ATE) presented in equation 5-1 will be employed for estimating the error in the robot trajectory. Illustration of the results obtained is given in Figure 5-10.

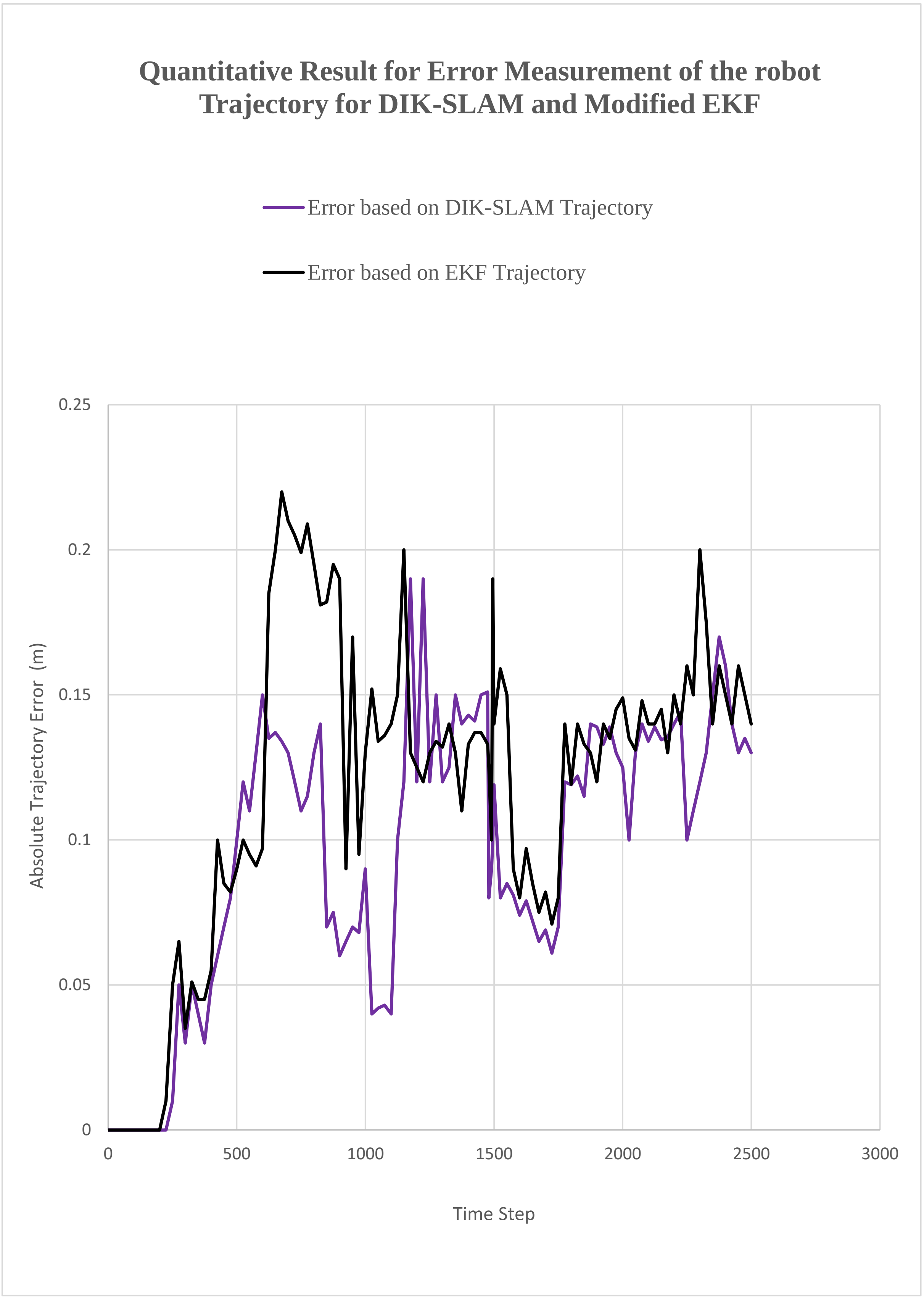

Figure 5-10: Quantitative error results for DIK-SLAM and modified EKF algorithm

In Figure 5-10, which represents the quantitative results of the error measurement for DIK-SLAM and EKF, the purple line colour in the graph in Figure 5-10 represents the result of the absolute trajectory error in the trajectory of DIK-SLAM and the black line colour in the graph represents the absolute trajectory error in the trajectory of EKF. Given Figure 5-10, the purple line colour in the graph which signifies the DIK-SLAM ,attained the lowest absolute trajectory error for all distance covered, while the black colour line which signifies EKF, attained the highest trajectory error. There are instances where the error value compared to the ground truth trajectory for both algorithms is high. This mostly happens at the corners, where misdetection appears a lot in the qualitative result. The trajectory error between DIK-SLAM and the modified EKF is relatively low for most trajectories. Unlike the original Monte-Carlo algorithm that will experience a low error margin at the beginning, but grows higher as more distance is covered towards the challenging part of the image before it eventually gets kidnapped. However, considering the relationship between error and accuracy (Kitchenham et al., 2001), all experiments carried out in this study support the DIK-SLAM attaining the lowest error in most trajectories, implying that DIK-SLAM attained the best accuracy in its trajectory compared to either original Monte-Carlo or the modified EKF.

5.10 Experiment 3: Computational Complexity as Related to Processing Time/Speed for SLAM Techniques

In this section, the SLAM techniques used for experimental purposes were validated based on their computational complexity as related to processing time. The computational complexity is a common issue complained by many researchers (Thrun et al., 2001, Abouzahir et al., 2014) and it was taken into consideration in this study. Therefore, it is important to measure the computational complexity of the DIK-SLAM technique for a comparison with the modified EKF, and Original MCL algorithm. In carrying out this experiment, a function in Matlab was employed to assist in measuring the processing time for each SLAM algorithm. These function are known as TIC and TOC (Knapp-Cordes and McKeeman, 2011). TIC is a function called before something is to be timed, and TOC is called afterwards. Using these functions, assists to estimate the processing time for running each algorithm, using TIC before running the program and TOC afterwards (Knapp-Cordes and McKeeman, 2011). Table 5.4 shows the results of the computational complexity as related to the processing time.

Table 5-4: Computational complexity as related to processing time

SLAM Algorithms	**Processing time (seconds)**
Monte-Carlo	$120_{(s)}$
DIK-SLAM	$140_{(s)}$
Modified EKF	$300_{(s)}$

In Table 5-4, the result obtained is similar to that of the publicly available dataset. The Monte-Carlo algorithm attained the lowest processing time, implying that Monte-Carlo algorithm has the fastest processing speed with the lowest computational cost, compared to other algorithms. The modified EKF has the highest processing time, implying that it suffers the highest computation cost with the slowest processing speed (Wei et al., 2019). This was complained by many researchers (Zhang et al., 2011, Thrun et al., 2001, Abouzahir et al., 2014) which influenced us in not selecting the EKF because, during re-modification of the algorithm, there is possibility of increasing its complexity more beyond acceptance, even though it would improve its robustness. Rather, we selected a particle-based technique. The DIK-SLAM algorithm has a processing time of 140s closer to the original Monte-Carlo than the modified EKF, but there is a possibility of improving the processing speed. This can be achieved by using other standalone filtering algorithms with lower computational complexity, than the one employed in this study or proposing a particle-based technique of addressing environmental noises rather than introducing new standalone filtering algorithms that also contribute to the DIK-SLAM technique complexity. These will be investigated in our next study.

5.11 Chapter summary

This chapter discussed the comparison of experiments that were implemented. The outcomes derived from them showed that the DIK-SLAM equipped with modified Monte-Carlo algorithm, attained better results than other SLAM algorithms, except for the results of experiment 8 with minimal difference in processing time, given the experiment of computational complexity as related to processing time. The results of the processing time for the DIK-SLAM for both private and public dataset compared to the fastest algorithm (original Monte-Carlo) were still acceptable, considering their accomplishment. However, it could be improved and more research is necessary to accomplish that goal

CHAPTER SIX

6 SUMMARY, CONCLUSION AND FUTURE WORK

This chapter presents the conclusion, summary and future work for the research conducted. The goal of the study was to implement a DIK-SLAM technique that would assist autonomous robots to attain perfect navigation when encountering problems such as dynamic environment, kidnapping, illumination variation and shadow in its environment. However, since we have accomplished this task, it is vital to discuss the SLAM technique achievement, to know whether the objective of the study was accomplished or not.

6.1 Summary of the study

The objectives and the research questions of the study were addressed by proposing a DIK-SLAM technique that was implemented by using Matlab. The results accomplished are validated by using Absolute Trajectory Error, Root Mean Squared Error (RMSE), TIC and TOC to measure the performance of the system. The accomplishment of the study is summarised in this chapter.

- The first objective that targeted previous studies conducted on SLAM techniques for identification of problems like dynamic algorithm, environmental noises, kidnap robot and concurrency algorithms have been addressed in chapters 2 and 3 which also provided a general review and discussion on SLAM and how it had evolved over the years.

- SLAM problems discovered in the literature review carried out in chapters 2 and 3 were as follows: the dynamic environment, kidnap problem and environmental noises. These problems were targeted and for them to be resolved, they needed the algorithms to be re-modified. Thus, The algorithm modifications were the second objective of the study, which was addressed and presented in detail in chapter 4.

- Given the situation of re-modification and the introduction of new algorithms to enhance the performance of a system, this practice increased the complexity as related to processing time. However, this issue was minimised by the use of concurrency technique, which was the third objective of the study. Thus, this was accomplished in chapter 4, which discussed various concurrency techniques, how they function and the reason for selecting the Rotating Tilling (RT) concurrency technique.
- The last objective of the study was the evaluation of the SLAM techniques performance and was carried in several experiments and a brief discussion thereof wasgiven below.
    - The experiment with the use of publicly available datasets was to test DIK-SLAM's ability to cope with publicly datasets. Thus, considering the result obtained, the RMSE for Absolute trajectory error and relative pose error translational of DIK-SLAM was lower for all experiments compared to the Monte-Carlo algorithm, except for the Freiburg2_360_kidnap experiment where the error results attained were higher. However, the reason for that were discussed in chapter 5. The results obtained were in accordance with the technique of RGB-D SLAM and RTAB proposed in the work of (Kaser, 2019) using a similar dataset.
    - The qualitative results in Figures 5-5a and 5-9 of the private dataset were evaluated quantitatively for the purpose of error measurement. The error measurement in the trajectory of the robot that was measured using the absolute trajectory error technique showed that in DIK-SLAM, the percentage overall error rate for all trajectory stood at 9% and in Monte-

Carlo algorithm, error stood at 20% while in modified EKF, error stood at 14%. Given this value for comparison, it showed that the DIK-SLAM attained minimal error than other SLAM algorithms used in this experiment.

- The qualitative results in Figures 5-5a and 5-8a had proven that DIK-SLAM could withstand the problem of kidnapping and dynamic environment respectively. However, if the noise intensity in Figure 5-3a was at a high rate, the DIK-SLAM could experience kidnapping as presented in Figure 5-5b. Thus, Measuring the qualitative results in Figure 5-8a, 81% detection of the dynamic environment was noted and presented in Table5-2, the results of which were accepted. However,, if the intensity of the noises were increased, we could have seen the dynamic detection performance value dropping below 81% with a lot of false detection, as it was presented in Table 5-3.
- The experiment for complexity computational cost as related to processing time had assisted us to realize a possible way of improving the DIK-SLAM complexity, which will be discussed under future work. The Monte-Carlo attained the fastest processing time for both public and private datasets, but the time differences between Monte-Carlo and DIK-SLAM algorithms were not much for both cases. Thus, considering the private dataset, the Monte-Carlo algorithm attained the fastest time of 120sec, while the DIK-SLAM was second in position with 140sec. The worst value was obtained from the modified EKF with a results that were more than double of DIK-SLAM processing time. However, the difference between the original Monte-Carlo and DIK-SLAM stood at

20sec, but considering the effectiveness and the number of problems DIK-SLAM can cope with, the current processing time was acceptable.

The experiment had proven DIK-SLAM performance and when benchmarked with the publicly available and private datasets, the DIK-SLAM still attained better performance than all SLAM algorithms employed for experimental comparison.

6.2 Summary of contribution

In the study conducted to present a novel SLAM technique, the following were contributed to the research.

- A similar DIK-SLAM technique that was presented in this study had been published by journals and conferences accredited by the Department of Higher Education and Training, South Africa.
- The DIK-SLAM had been evaluated for trajectory error and from the results obtained, if the system is adopted by industries and companies involved in exploration, it would increase productivity while reducing accidents and injuries.
- The effect of environmental noises, dynamic environment and kidnaping that cause failure to SLAM technique had been minimized and overcame in this study.
- The high computation complexity as related to processing time after introducing filtering algorithms had been minimized to some extent.
- A good comparison of computational complexity as related to processing time/speed for various algorithm is presented in this study.

6.3 Conclusion

The attention on SLAM is increasing daily because it supports the possibilities of concurrent execution of mapping and localization processes. These become a great achievement in solving the problem of mobile robot autonomously, achieving its goal without being controlled by anyone. However, this accomplishment has not been fully attained due to various limitations of SLAM. The previous review conducted for this study showed, that SLAM still suffers from the issue of illumination variation, kidnap robot, high computational cost and dynamic environment (Agarwal and Burgard, 2015., Irie et al., 2012., Jia et al., 2016. .). The proposed DIK-SLAM technique has the ability to overcome the problem of illumination variation due to the presence of filters introduced into the system. These filters are operating in parallel because the computational cost as related to processing speed was also taken into consideration. The DIK-SLAM technique was also equipped with a modified Monte-Carlo algorithm to cope with a dynamic environment, based on the cell occupancy technique. The DIK-SLAM technique can also handle the kidnap problem by using the initializing localization technique.

Matlab was used for implementation and the experiment was validated based on absolute trajectory error, Root Mean Squared Error (RMSE) and computational cost as related to processing time. The graphical representation result showed better performance of DIK-SLAM compared to other SLAM algorithms when benchmarked with the publicly available and private datasets. Thus, the original Monte-Carlo attained the best result for less computational complexity, but the result attained by the DIK-SLAM was acceptable considering the number of problems the technique was addressing. Eventhough success was attained in this study, the next important task is on how to further reduce the computational cost to attain maximum processing speed and how to improve its effectiveness around the corners. Although we have an idea that will be discussed briefly under future work, but more research is necessary to be carried out.

Considering the experiment carried out in this study, it can be concluded that the DIK-SLAM equipped with the modified Monte-Carlo algorithm and filters has the ability to cope with the issue of Illumination variation, dynamic environment and kidnapping at a lower computation complexity than the modified EKF to attain perfect navigation in its environment.

6.4 Future work

Over the years, the SLAM technique has been proposed to address navigations for an autonomous robot, but has been confronted with many challenges. However, some of these challenges are addressed in this study, but there are limitations that still need to be accomplished to present a future SLAM technique better than DIK-SLAM. For instance, the study only targeted two environmental noises (shadow and light intensity). There are several environmental noises like humidity, mist, fog, and snow, that we did not address in this study, which are still challenges that need to be resolved in the SLAM technique.

The filtering stage is also an area where many challenges still exist. Apparently, this stage contributes to an increase in computational complexity. This issue will be targeted in the future study by proposing a particle-based technique to overcome the issue of environmental noises, rather than introducing standalone filters that increase the computation complexity. The research study also targeted the Kidnap problem, which will only be successful if the dissimilarities in the images are low and kidnapping is not happening frequently. Otherwise, the robot might be unable to recover from kidnapping just as it was presented in Figure 5-5b when the noise intensity was increased at a high rate and DIK-SLAM could not recover from kidnapping. Therefore, more research is necessary to handle cases when the dissimilarity is high in images and frequent occurrence of kidnapping in the environment scene.

The study also targeted the issue of the dynamic environment but could perform better when moving objects are not too much in the environment. Alternatively, another technique is to introduce the Multi Tracking Object and Detection algorithm into MCL to enhance its ability to track multiple moving objects and dynamic environments, but with an increase in system complexity that suffers from high computational cost than our technique (Moratuwage et al., 2013). Given that the study targeted to reduce the computational cost, we could not propose to introduce the Multi Tracking Object and Detection algorithm into MCL, instead, the cell occupancy technique, based on the assumption that cells can change independently, was proposed in this study. because it works better with a low computational cost.

However, this assumption might not be accurate in all instances. Thus, the group of adjacent cells might be occupied with the same object violating the assumption, because cells might be dependent on each other with an effect that would minimize system performance. In forthcoming, we plan to propose a better assumption to improve the performance of the system.

Furthermore, the corners pose challenges for DIK-SLAM and other SLAM algorithms used in this study. It was at these corners, the SLAM algorithms used for experiment experienced higher error in their trajectories. Thus, for future studies, learning algorithms that can handle corners could be introduced into SLAM to overcome the challenges of the corners, but more research is still necessary so as to understand how the learning algorithm could be integrated into SLAM.

Lastly, the proposed SLAM stages employed in the study, were not only adaptive to our algorithm, but researchers could also still investigate this stage by introducing different SLAM algorithms for comparison to reveal the system with the greatest performance.

The most challenging issue is acquiring 100% accuracy in trajectory under any circumstances and this has not been achieved at the moment but for this study based on the dynamic noisy environment, 89% trajectory accuracy was obtained. Therefore, there is still need for further investigation in SLAM research.

6.5 Chapter summary

This study has proposed a DIK-SLAM technique by using filtering algorithms and concurrency technique for minimizing the issue caused by environmental noises and re-modified Monte-Carlo to overcome the issue of dynamic environment and kidnapping problem. The experiment was carried out by using various evaluation schemes and outcomes showed a satisfactory performance of the DIK-SLAM technique.

REFERENCES

1. ABOUDAYA, E., DURAIRAJ, A., LONGINO, J. & YINKFU, K. Lecture 4: Image enhancement using median filtering. *Machine Vision and Digital Analysis,* 2006.
2. ABOUZAHIR, M., ELOUARDI, A., BOUAZIZ, S., LATIF, R. & TAJER, A. FastSLAM 2.0 running on a low-cost embedded architecture. *13th International Conference on Control Automation Robotics & Vision (ICARCV),* 10-12 December, pages 1421-1426, 2014.
3. AGARWAL, P. & BURGARD, W. Robust graph-based localization and mapping, PhD thesis, *University of Freiburg, Germany***,** pages 1-155, 2015.
4. AGHA-MOHAMMADI, A. A., AGARWAL, S., CHAKRAVORTY, S. & AMATO, N. M. Simultaneous localization and planning for physical mobile robots via enabling dynamic replanning in belief space. *Robotics: Systems and Control***,** pages 1-21, 2015.
5. AGUNBIADE, O. Y., NGWIRA, S. M. & ZUVA, T. 2014. Enhancement optimization of drivable terrain detection system for autonomous robots in light intensity scenario. *The International Journal of Advanced Manufacturing Technology,* vol 74**,** pages 629-636, 2014.
6. AGUNBIADE, O. Y., ZUVA, T., JOHNSON, A. O. & ZUVA, K. Enhancement performance of road recognition system of autonomous robots in shadow. *Signal & Image Processing: An International Journal.,* vol 4, issue 6, 2013.
7. AL-AMRI, S. S., KALYANKAR, N. V. & KHAMITKAR, S. D. A comparative study of removal noise from remote sensing image. *International Journal of Computer Science Issues, (IIJCS),* vol 7, issue 1**,** pages 32-36, 2010.
8. AL-MUTIB, K. N. SLAM based autonomous mobile robot navigation using stereo vision. *International Journal of Computer Science and Network,* vol 4**,** pages 873- 886, 2015.
9. ALQADI, Z. Salt and pepper noise: effects and removal. *International Journal on Electrical Engineering and Informatics,* vol 2, 2018.
10. ANGELI, A., DONCIEUX, S., MEYER, J. & FILLIAT, D. Real-time visual loop-closure detection. *IEEE International Conference on Robotics and Automation,* 19-23 May, pages 1842-1847, 2008

11. AVOTS, D., LIM, E., THIBAUX, R. & THRUN, S. A probabilistic technique for simultaneous localization and door state estimation with mobile robots in dynamic environments. *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 30 September - 4 October, vol 1, papes 521-526, 2002.
12. BAEZA-YATES, R. & LIAGHAT, Z. Quality-efficiency trade-offs in machine learning for text processing. *IEEE International Conference on Big Data (Big Data),* 11-14 December. pages 897-904, 2017.
13. BAILEY, T., NIETO, J., GUIVANT, J., STEVENS, M. & NEBOT, E. Consistency of the EKF-SLAM ALgorithm. *Australian Centre for field Robot,* pages 1-20, 2014.
14. BOUTHEMY, P. A maximum likelihood framework for determining moving edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol 11**,** pages 499-511, 1989.
15. BOWMAN, S. L., ATANASOV, N., DANIILIDIS, K. & PAPPAS, G. J. Probabilistic data association for semantic SLAM. *IEEE International Conference on Robotics and Automation (ICRA)***,** pages 1722-1729, 2017.
16. BUKHORI, I. & ISMAIL, Z. H. Detection of kidnapped robot problem in Monte-Carlo localization based on the natural displacement of the robot. *International Journal of Advanced Robotic Systems. SAGE***,** pages 1-6, 2017.
17. BYRON, B. & GEOFFREY, J. G. A spectral learning approach to range-only SLAM. *International Conference on Machine Learning, Atlanta, Georgia,* vol 28, pages 1-19, 2013.
18. CARLONE, L., NOG, M. K., DU, J., BONA, B. & INDRI, M. Simultaneous localization and mapping using Rao-Blackweillized particle filters in multi robot system. *Journal of Intelligent Robot system,* vol 63**,** pages 283-307, 2011.
19. CASTELLANOS, J. A., NEIRA, J. & TARDOS, J. D. Multisensor fusion for simultaneous localization and map building. . *IEEE Transactions on Robotics and Automation.,* vol 17**,** pages 908-914, 2001.
20. CERIANI, S., FONTANA, G., GIUSTI, A., MARZORATI, D., MATTEUCCI, M., MIGLIORE, D., RIZZI, D., SORRENTI, D. G. & TADDEI, P. Rawseeds ground truth collection systems for indoor self-localization and mapping. *Autonomous Robots,* vol 27, 2009.

21. CHANGCHANG, W., CLIPP, B., XIAOWEI, L., FRAHM, J. M. & POLLEFEYS, M. 3D model matching with viewpoint-invariant patches. *IEEE Conference on Computer Vision and Pattern Recognition*, 23-28 June, pages 1-8, 2008.
22. CHEN, J. & LUM, K. Simultaneous localization and mapping based on particle filter for sparse environment. *International Conference on Mechatronics and Control (ICMC)*, 3-5 July, pages 1869-1873, 2014.
23. CHEN, Y. Algorithm for Simultaneous localization and mapping. pages 1-15, 2013.
24. CHIN, F. L., DON, L. Y. & YEH, C. C. A rotate-tilling image composition method for parallel volume rendering on distribution memory multicomputers. *Feng Chia University, Taichung, Tiawan*. 2001.
25. CHOI, J. & MAURER, M. local volumetric hybrid-map-based simultaneous localization and mapping with moving object tracking. *IEEE Transactions on Intelligent Transportation Systems,* vol 17**,** pages 2440-2455, 2016.
26. CHUHO, Y. & BYUNG-UK, C. Detection and recovery for kidnapped-robot problem using measurement entropy. *Springer-Verlag Berlin Heidelberg,* vol 261**,** pages 293-299, 2011.
27. CLIPP, B., ZACH, C., LIM, J., FRAHM, J.-M. & POLLEFEYS, M. Adaptive, real-time visual simultaneous localization and mapping. *Department of computer Science, University of North Caroline,* pages 1-8, 2009
28. CRAS, J. L., PAXMAN, J. & SARACIK, B. Vision based localization under dynamic illumination. *5th International Conference on Automation, Robotics and Applications (ICARA)***,** pages 453-458, 2011.
29. DELLAERT, F. Factor graphs and GTSAM: A hands-on introduction. *Technical Report number GT-RIM-CP&R-2012-002***,** pages 1-27, 2012
30. DELLAERT, F., CARLSON, J., ILA, V., NI, K. & THORPE, C. E. Subgraph-preconditioned conjugate gradients for large scale SLAM. *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 18-22 October, pages 2566-2571, 2010.

31. DEMING, R. & PERLOVSKY, L. Concurrent detection and tracking using multiple flying sensors. *4th IEEE Workshop on Sensor Array and Multichannel Processing*, 12-14 July, pages 505-509, 2006.
32. DISSANAYAKE, G., HUANG, S., WANG, Z. & RANASINGHE, R. A review of recent developments in simultaneous localization and mapping. *6th International Conference on Industrial and Information Systems*, 16-19 August, pages 477-482, 2011.
33. DJEKOUNE., O., ACHOUR, K. & TOUMI, R. A sensor based navigation algorithm for a mobile robot using the dvff approach. *International Journal of Advanced Robotic Systems*, 2009.
34. ELIAZAR, A. & PARR, R. DP-SLAM: fast, robust simultaneous localization and mapping without predetermined landmarks. *Proceedings of the 18th international joint conference on Artificial intelligence. Acapulco, Mexico: Morgan Kaufmann Publishers Inc, 2003.*
35. ELIAZAR, A. I. & PARR, R. Constant/linear time simultaneous localization and mapping for dense maps. 2019
36. ENGEL, J. S. & CREMERS, D. Large-scale direct SLAM with stereo cameras. *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference*, pages 1935-1942, 2015.
37. FERNÁNDEZ-MADRIGAL, J. A. & CLARACO, J. L. B. Simultaneous localization and mapping for mobile robots: Introduction and methods, *Information Science Reference*, 2013.
38. FUENTES-PACHECO J, J., RUIZ-ASCENCIO, REND, J. M., & N-MANCHA. Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev*, vol 43, 2015.
39. GANAPATHY, V., YUN, S. C. & CHIEN, T. W. Enhanced D* lite algorithm for autonomous mobile robot. *School of Engineering, Monash University Sunway Campus,* vol 1**,** pages 58-73, 2011.
40. GAO, X. & ZHANG, T. Robust RGB-D simultaneous localization and mapping using planar point features. *Robot. Auton. Syst.*, vol 72**,** pages 1-14, 2015
41. GOMEZ-OJEDA, R., MORENO, F.-A., SCARAMUZZA, D. & GONZALEZ-JIMENEZ, J. PL-SLAM: A Stereo SLAM system through the combination of points and line segments. *The*

*Machine Perception and Intelligent Robotics (MAPIR) Group, University of Malaga***,** pages 1-11, 2017.

42. GUYONNEAU, R., LAGRANGE, S., HARDOUN, L. & LUCIDARME, P. The kidnapping problem of mobile robots: A set membership approach. *7th National Conference on Control Architectures of Robots*, 2012
43. HADJI, S. E., KAZI, S. & HING, T. H. A Review: Simultaneous localization and mapping algorithms for mobile robot. *1st International Conference of Recent in Informtion and Communication Technologies***,** pages 83-92, 2014
44. HESH, J. & TRAWNY, N. Simultaneous localization and mapping using an omni-directional camera. *CSCI 556--Computer Vision,* pages 1-10, 2005.
45. HIRYU, S., BATES, M. E., SIMMONS, J. A. & RIQUIMAROUX, H. FM echolocating bats shift frequencies to avoid broadcast–echo ambiguity in clutter. *Proceedings of the National Academy of Sciences.*, vol 107, papes 7048-7053, 2010
46. HO, K. L. & NEWMAN, P. Detecting loop closure with scene sequences. *International Journal of Computer Vision,* vol 74, 261-286, 2007.
47. HUANG, J., KONG, B., LI, B. & ZHENG, F. A new method of unstruncture road detection based on hsv color space and road features. *Proceeding of IEEE International conference Information Acquisition, 2007.*
48. HUANG, Y. L. & FUH, C. S. Noise reduction using enhanced bilateral filter. *Department of Computer Science and Information Engineering, National Taiwan University, Taiwan,* vol 12, no 4**,** pages 46-53, 2006.
49. IRIE, K., YSHIDA, T. & TOMONO, M. 2012. Outdoor localization using stereo vision under various illumination conditions. *Advanced Robotics,* vol 26, pages 327-348, 2012.
50. ISHIGURO, H. & TSUJI, S. Image-based memory of environment. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 8-9 November, vol 2, pages 634-639, 1996.
51. JACK, T., JESSICA, K. H. & BRIAN, K. G. Two methods for display of high contrast images. *ACM Trans. Graph.,* vol 18**,** pages 56-94, 1999.

52. JEAN-ARCADY, M. & DAVID, F. Map-based navigation in mobile robots: A review of map-learning and path-planning strategies. *Cognitive Systems Research,* vol 4**,** pages 283-317, 2003

53. JIA, S., WANG, K. & LI, X. Mobile robot simultaneous localization and mapping based on a monocular camera. *Journal Robot.,* vol 2, 2016

54. KAESS, M., RANGANATHAN, A. & DELLAERT, F. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics,* vol 24, pages 1365-1378, 2008

55. KASER, A. benchmarking and comparing popular visual slam algorithms. *Asian Journal of Convergence In Technology,* vol 5, Issue 1**,** pages 1- 7, 2019

56. KHAN, S. A., CHOWDHURY, S. S., NILOY, N. R., AURIN, F. T. Z., AHMED, T. & MOSTAKIM, M. Simultaneous localization and mapping (SLAM) with an autonomous robot. *Department of Electrical and Electronic Engineering, School of Engineering and Computer Science, BRAC University***,** pages 1-41, 2018.

57. KIRCHNER, M. & FRIDRICH, J. On detection of median filtering in digital images. *Technische University Dresden, Department of Computer Science, Germany,* 2010.

58. KITCHENHAM, B. A., PICKARD, L. M., MACDONELL, S. G. & SHEPPERD, M. J. What accuracy statistics really measure [software estimation]. *IEEE Proceedings - Software,* vol 148**,** pages 81-85, 2001.

59. KNAPP-CORDES, M. & MCKEEMAN, B. Improvements to tic and toc functions for measuring absolute elapsed time performance in MATLAB. *The MathWorks, Inc. MATLAB and Simulink***,** pages 1-4, 2011.

60. KONOLIGE, K., GRISETTI, G., KÜMMERLE, R., BURGARD, W., LIMKETKAI, B. & VINCENT, R. Efficient sparse pose adjustment for 2D mapping. *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 18-22 October, pages 22-29, 2010.

61. KRAJSEK, K. & MESTER, R. The edge preserving wiener filter for scale and trensor valued images. *J. W. Goethe University, Frankfurt, Germany*, 2004

62. KRISHNA KANT SINGH, PAL, K. & NIGAM, M. J. Shadow detection and removal fro remote sensing images using NDI and Morphological operation. *International Journal of Computer Applications,* vol 42, no 10**,** pages 37-40, 2012.

63. LUIS, P., LORENZO, F., ARTURO, G. & OSAR, R. Map building and monte-carlo localization using global apperance of ominidirectional images. *Sensors (PubMed),* vol 10, issue 12**,** pages 11468-11497, 2010.

64. LANG, Z., MA, X. & DAI, X. Simultaneous planning localization and mapping in a camera network. vol 24, pages 1037-1058, 2010.

65. LE CRAS, J., PAXMAN, J. & SARACIK, B. Improving robustness of vision based localization under dynamic illumination. *Recent Advances in Robotics and Automation, Berlin, Heidelberg: Springer,* 2013

66. LEE, S., LEE, S. & KIM, D. Recursive unscented kalman filtering based slam using a large number of noisy observations. *International Journal of Control Automation and Systems,* vol 4, no 6, pages 736-747, 2006

67. LI-CHEE-MING, J. & ARMENAKIS, C. Augmenting ViSP's 3D model-based tracker with RGB-D SLAM for 3D pose estimation in indoor environments. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.,* pages 925-932, 2016

68. LIN, C. F., CHUNG, Y. C. & YANG, D. L. Efficient parallel volume rendering methods on distributed memory multicomputers. *Department of Engineering and computer science, Feng Chia University , Taichung, Taiwan 407,* 2003.

69. LIN, L.-H., LAWRENCE, P. D. & HALL, R. Robust outdoor stereo vision SLAM for heavy machine rotation sensing. *Machine Vision and Applications,* vol 24, pages 205-226, 2013

70. LUIS, M., PABLO, B., PILAR, B. & .JOSE, M. Multi-cue visual obstacle detection for mobile robots. *Journal of physical agents.,* vol 4, issue 1, 2010.

71. MAKHUBELA, J. K., ZUVA, T. & AGUNBIADE, O. Y. Framework for visual simultaneous localization and mapping in a noisy static environment. *International Conference on Intelligent and Innovative Computing Applications (ICONIC),* 6-7 December, pages 1-6, 2018.

72. Martinet, P., Gallice, J. & Khadraoui, D. Vision based control law using 3D visual feaures. http://www.researchgate.net/publiction/2380398_vision_Based_control_law_using_3D_visual_features, 1997

73. MATEI, B. C., VALK, N. V., ZHU, Z., CHENG, H. & SAWHNEY, H. S. Image to LIDAR matching for geotagging in urban environments. *IEEE Workshop on Applications of Computer Vision (WACV)*, 15-17 January, pages 413-420, 2013.

74. MILSTEIN, A. Occupancy grid maps for localization and mapping. *University of waterloo***,** pages 381-411, 2008.

75. MING, W. & JI-YING, S. Simultaneous localization, mapping and detection of moving objects with mobile robot in dynamic environments. *2nd International Conference on Computer Engineering and Technology*, 16-18 April, 2010.

76. MONTEMERLO, M. & THRUN, S. Simultaneous localization and mapping with unknown data association using FastSLAM. *IEEE International Conference on Robotics and Automation*, 14-19 September. vol 2, pages 1985-1991, 2003.

77. MONTEMERLO, M., THRUN, S., KOLLER, D. & WEGBREIT, B. FastSLAM: a factored solution to the simultaneous localization and mapping problem. *Eighteenth national conference on Artificial intelligence. Edmonton, Alberta, Canada: American Association for Artificial Intelligence, 2002.*

78. MORATUWAGE, D., VO, B. & WANG, D. Collaborative Multi-vehicle SLAM with moving object tracking. *IEEE International Conference on Robotics and Automation,* 6-10 May, pages 5702-5708, 2013.

79. MU, B. Value of Information based distributed inference and planning. *PhD Thesis, Massachusetts Institute of Technology*, 2013.

80. NEGENBORN, R., JOHANSEN, P. P. & WIERING, M. Robot localization and kalman filter. *Institute of Information and Computing Sciences*, 2003.

81. NEWMAN, P. & KIN, H. SLAM-loop closing with visually salient features. *Proceedings of the IEEE International Conference on Robotics and Automation*, 18-22 April, pages 635-642, 2005.

82. NEZAMPOUR, A., NASRI, A. & SCHOBER, R. Asymptotic analysis of space-time codes in generalized keyhole fading channels. *IEEE Transactions on Wireless Communications,* vol 10**,** pages 1863-1873, 2011

83. NOBELEN, R. V. & TAYLOR, D. P. Analysis of the pairwise error probability of noninterleaved codes on the Rayleigh-Fading channel. *IEEE Transactions on Communications,* vol 44**,** pages 456-463. 1996

84. OH, S., HAHN, M. & KIM, J. Simultaneous localization and mapping for mobile robots in dynamic environments. *International Conference on Information Science and Applications (ICISA),* 24-26 June, pages 1-4, 2013.

85. OH, T., LEE, D., KIM, H. & MYUNG, H. Graph structure-based simultaneous localization and mapping using a hybrid method of 2d laser scan and monocular camera image in environments with laser scan ambiguity. *Sensors,* vol 15**,** 2015.

86. PARKER, S., PARKER, M., LIVNAT, Y., SLOAN, P.-P., HANSEN, C. & SHIRLEY, P. Interactive ray tracing for volume visualization. *IEEE Transactions on Computer Graphics and Visualization***,** pages 1-13, 2001.

87. QIU, C., ZHU, X. & ZHAO, X. Vision-based unscented FastSLAM for mobile robot. *Proceedings of the 10th World Congress on Intelligent Control and Automation,* 6-8 July, pages 3758-3763, 2012

88. SALEEM, M. M. An economic simultaneous localization and mapping system for remote mobile robot using SONAR and an innovative AI algorithm. *International Journal of Future Computer and Communication,* vol. 2, no 2**,** pages 147-150, 2013.

89. SANO, K., KITAJIMA, H., KOBAYASHI, H. & NAKAMURA, T. Data-parallel volume rendering with adaptive volume subdivision. *The Institute of Electronics, Information and Communication Engineers,* pages 80-89, 2000.

90. SARANYA, C., UNNIKRISHNAN, M., ALI, S. A., SHEELA, D. S. & LALITHAMBIKA5, V. R. Terrain based D* algorithm for path planning. *International Federation of Automatic Control***,** pages 178-182, 2016

91. SE, S., LOWE, D. & LITTLE, J. Vision-based mobile robot localization and mapping using scale-invariant features. *Proceedings IEEE International Conference on Robotics and Automation (ICRA),* vol 2, pages 2051-2058, 2001

92. SHALAL, N., LOW, T., MCCARTHY, C. & HANCOCK, N. Orchard mapping and mobile robot localisation using on-board camera and laser scanner data fusion – Part a: Tree detection. *Computers and Electronics in Agriculture,* vol 119**,** pages 254-266, 2015.

93. SHENGYAN, Z. & KARL, L. Self- supervised learning method for unstructured road detection using fuzzy support vector machines. *Proceeding of Robotic Mobility Group. Massachusetts Institute of Technology, Boston, USA*, 2010.

94. SHIGUANG, W., MINGDE, Y., CHENGDONG, W. & JUN, L. An improved FastSLAM2.0 algorithm based on ant colony optimization. *29th Chinese Control And Decision Conference (CCDC)*, 28-30 May, pages 7134-7137, 2017.

95. SHUANG, N., YUANYUAN, S., HUI, D. & ZHONG, L. Improvement of low illumination image enhancement algorithm based on physical mode. *BioTechnology: An Indian Journal,* vol 10, Issue 22**,** pages 13995-14001, 2014.

96. SINECEN, M. Digital image processing with MATLAB. 2016.

97. SINGH, Y. & GOYAL, E. R. A study of various haze removal algorithm. *International Journal of Innovative Research in Technology,* vol 1**,** pages 1-7, 2014.

98. SKRZYPCZY, P. Simultaneous localization and mapping: A feature-based probabilistic approach. *Int. J. Appl. Math. Computer. Sci.,* vol 19**,** pages 575-588, 2009.

99. SOUNDERHAUF, N., PROTZEL, P. & SUCHY, J. Robust optimization for simultaneous localization and mapping. *Doctoral Thesis submitted to the Faculty of Eletronice and Information science: Technische Universitat Chemnitz***,** pages 1-193, 2012.

100. STACHNISS, C., GRISETTI, G., HAHNEL, G. & BURGARD, W. Imrpoved Rao-Blackwellized mapping by adaptive sampling and active loop-closure. *In Proc. of the Workshop on Self-Organization of Adaptive Behaviour***,** pages 1-15, 2004.

101. STECKEL, J. & PEREMANS, H. BatSLAM: Simultaneous localization and mapping using biomimetic sonar. *Plos one.,* vol 8, pages 1-11, 2013.

102. STURM, J., ENGELHARD, N., ENDRES, F., BURGARD, W. & CREMERS, D. A benchmark for the evaluation of RGB-D SLAM systems. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 7-12 October, pages 573-580, 2012.

103. TAN, F., LOHMILLER, W. & SLOTINE, J. E. Simultaneous localization and mapping without linearization. *Massachusetts Insitute of Technology,* pages 1-50, 2015.

104. TAN, W., ZILONG, H. L., GUOFENG, D. & BAO, Z. H. Robust monocular SLAM in dynamic environments. *IEEE International Symposium on Mixed and Augmented Reality Science and Technology Proceedings***,** pages 209-218, 2013.

105. TENG, F. C. Real-time control using Matlab simulink. *IEEE international conference on systems, man and cybernetics. 'cybernetics evolving to systems, humans, organizations, and their complex interactions*, 8-11 October, pages 2697-2702 vol.4, 2000.

106. THAMRIN, N. M., ADNAN, R. A., SAM, R. & RAZAK, N. Simultaneous localization and mapping based real-time inter-row tree tracking technique for unmanned aerial vehicle. 2012.

107. THRUN, S., FOX, D., BURGARD, W. & DELLAERT, F. Robust Monte-Carlo localization for mobile robots. *Artificial Intelligence,* vol 128, pages 99-141, 2001.

108. TIAN, Y. & MA, S. Probabilistic double guarantee kidnapping detection in SLAM. *Robotics and Biomimetics,* vol 3**,** 2016.

109. TIAN, Y. & MA, S. 2017. Kidnapping Detection and recognition in previous unknown environment. *Journal of Sensors,* pages 1-15, 2017

110. WALCOTT-BRYANT, A., KAESS, M., JOHANNSSON, H. & LEONARD, J. J. Dynamic pose graph SLAM: Long-term mapping in low dynamic environments. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 7-12 October, pages 1871-1878, 2012

111. WANG, Y., CHEN, W. & WANG, J. Map-based localization for mobile robots in high-occluded and dynamic environments. *Industrial Robot: An International Journal,* vol 41, no 3 pages 241–252, 2014

112. WEI, Z., LIU, S., SUN, Y. & LING, H. Accurate facial image parsing at real-time speed. *IEEE Transactions on Image Processing,* vol 28**,** pages 4659-4670, 2019.

113. WEN, Q., YANG, Z., SONG, Y. & JIA, P. Road boundary detection in complex urban environment based on low-resolution vision. *proceedings of the 11th joint conference on information Science, State Key Laboratory on Intelligent Technology and Systems, Tsinghua*

*National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University Beijing. 100084, China***,** pages 1-7, 2008

114. WURM, K. M., STACHNISS, C., GRISETTI, G. & BURGARD, W. Improved simultaneous localization and mapping using a dual representation of the environment. *Department of Computer Science, University of Freiburg, Germany***,** pages 1-10, 2003.
115. YENIKAYA, S., YENIKAYA, G. & DUVEN, E. Keeping the vehicle on the road- A survey on-road lane detection system. *ACM Computing Surveys. Uludag University, Turkey,* vol 46**,** pages 1-43, 2013.
116. YI, Y. & WANG, Z. Robot localization and path planning based on potential field for map building in static environments. *An International Journal for the Publication of Original Research on the Analysis of Structures, Materials and New Technologies in the Field of Mechanical Engineering, Naval Architecture, Basic Technical Sciences, Electrical Engineering, Computing and Civil Engineering,* vol 35**,** pages 171-178, 2015.
117. YINKA, A. O., NGWIRA, S. M., TRANOS, Z. & SENGAR, P. S. Performance of drivable path detection system of autonomous robots in rain and snow scenario. *International Conference on Signal Processing and Integrated Networks (SPIN),* 20-21 Feburary, pages 679-684, 2014.
118. ZANETTI, R. 2012. Recursive update filtering for nonlinear estimation. *IEEE Transactions on Automatic Control,* vol 57**,** pp 1481 - 1490.
119. ZEHANG, S., BEBIS, G. & MILLER, R. On-road vehicle detection: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol 28**,** pages 694-711, 2006
120. ZHANG, H. & LI, M. Rapid path planning algorithm for mobile robot in dynamic environment. *Advances in Mechanical Engineering,* vol 9**,** 2017.
121. ZHANG, J., SUN, J., JIN, Z., ZHANG, Y. & ZHAI, Q. Survey of parallel and distributed volume rendering: Revisted *International Conference of Computational Science and it Application***,** pages 435-444, 2015.
122. ZHANG, L., ZAPATA, R. & LEPINAY, P. Self-adaptive Monte-Carlo localization for mobile robots using range finders. *Robotica,* vol 30**,** pages 229-244, 2011

123. ZHANG, Z., LIU, S., TSAI, G., HU, H., CHU, C.-C. & ZHENG, F. PIRVS: An advanced visual-inertial SLAM system with flexible sensor fusion and hardware co-design. *PerceptInc., Santa Clara, CA 95054, USA,*2017.

124. ZOU, B., ZHANG, X., LIAO, S. & WANG, L. Specularity removal using dark channel prior. *Journal of Information Science and Engineering*, vol 29, pages 835-851, 2013.

125. ZUNINO, G. & CHRISTENSEN, H. I. Simultaneous localization and mapping in domestic environments. *International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 67-72, 2001.