Heriot-Watt University

School of Engineering and Physical Sciences

MSc in Robotics, Autonomous and Interactive Systems

Dissertation

Title:	Investigating path planning control of an energy-efficient wheeled autonomous mobile robot
Author:	Nurgaliyev Shakh-Izat
Registration Number	H00226564
Date	26.08.2016
Supervisor:	Xianwen Kong

DECLARATION OF AUTHORSHIP

I, Nurgaliyev Shakh-Izat, at this moment confirm that this thesis “**Investigating path planning control of an energy-efficient wheeled autonomous mobile robot**” submitted for assessment is my unaided work. All references have been quoted, and all sources of information, including graphs, data sets, ideas, equations, figures, and tables have been duly acknowledged at any point of their use.

This thesis was not previously showed to another examination board and has not been published.

Date: 26.08.2016 Signature: [illegible]

ABSTRACT

This thesis investigates and describes path planning control of an energy-efficient wheeled autonomous mobile robot.

The key problem of mobile robotics is to solve navigation problems, which is determining the position of the mobile robot in the workspace - the localization, and descriptions of the world – mapping [4]. Another issue is an energy limitation [1]. It is one of the most important challenges for mobile robots. Mobile robots have limited energy and computing power, so the applied methods and algorithms have to work in low-resource conditions.

The challenge of global navigation was solved by an algorithm of particle filter [2]. This method allows improving the accuracy of determining the position of the robot and the quality of the map.

In the thesis was considered a two-wheeled mobile robot, which is controlled by specifying the linear and angular velocity, which can be converted into the speed of the left and right wheels. Information about the outside world, the mobile robot receives from the laser scanning rangefinder.

The results show that motion of mobile robot consumes 50% power on average. Therefore, it is important to consider the energy-efficient designs and make mobile robot that consumes less energy.

Developed mobile localization and robot control methods have been tested experimentally. Experimental work on a real mobile robot equipped with a laser rangefinder conducted in the center of the "Robotics" of Heriot-Watt University.

ACKNOWLEDGEMENTS

Let me express my deep gratitude to all those who participated in the preparation, presentation, and discussion of my dissertation.

First of all, I sincerely grateful to Heriot-Watt University (United Kingdom) and its staff for the attention shown to my thesis. Heriot-Watt University provided all materials and facilities.

I would like to express my sincere appreciation and gratitude to my supervisor Xianwen Kong for assistance at all stages of the thesis.

Finally, I wish to thank my parents and friends for their support and assistance during my study.

CONTENTS

LIST OF FIGURES

ABBREVIATIONS

SLAM	Simultaneous Localization and Mapping
GPS	Global Positioning System
IMU	Inertial Measurement Unit
MIT	Massachusetts Institute of Technology
V-REP	Virtual Robot Experimentation Platform
LIDAR	Light Detection and Ranging
GUI	Graphical user interface

Chapter 1

Introduction

1.1 RATIONALE FOR THE STUDY

In the 21st century, mobile robotic systems commonly integrated into a variety of industries. Modern robotics in general and mobile robots, in particular, are developing by leaps and bounds. Mobile robots are used successfully in all spheres of human activity (as industrial assembly machines, surgical devices in medicine and so on) [3].

Most of the tasks that are put in front of mobile robots, associated with the generating of certain sequences of actions necessary to achieve the objective. In the case of navigation tasks, these steps can be formalized regarding achieving a robot in certain areas of space. Path planning control or navigation(locomotion) of the mobile robot is a widely-studied and urgent task [4]. Although there are known ways to solve this problem in the case of a single goal in the environment, when you try to extend these techniques to a multi-purpose case, it creates some difficulties.

Progress achieved through the improvement of the actuators and mainly sensory systems. The main tendency of the modern mobile robot is to move from remote-controlled systems, which require direct human intervention to autonomous systems in which the operator specifies only the final goals.

Mobile robot must meet the following requirements [4]:

- To provide automatic movement in the plane
- To determine its position in space
- To create a map of working area

Meeting these requirements is possible with the help of modern equipment of mobile robots by various sensors: odometers, satellite navigation systems, inertial measurement systems, video cameras, sonars and laser scanning rangefinders. A wide variety of sensor systems and at the same time the growth of computing power of onboard control systems allow researchers to develop different methods of sensory information processing for solving the problems of navigation and control of mobile robots [3].

The key problem of mobile robotics is to solve navigation problems, implying the determining the position of the mobile robot in the workspace - the localization, descriptions of the world – mapping [4]. Information about the current position of the robot is required to solve the majority of problems: the passage of a predetermined path, the path to the specified point, return to the starting position. Information about the world can be represented as a map or plan of the area.

Most of the existing methods of navigation do not provide high accuracy of localization. Because of this, there is huge error localization in space, which do not allow to achieve the goal of navigation successfully. Despite the considerable amount of research in this area, the problem of navigation and finding the optimal parameters for speed, cost, energy efficiency is still open.

Research in the field of development of methods of navigation and control of mobile robots are conducted in many research centers of Great Britain - as well as in US universities - Carnegie Mellon University, Stanford University Germany - University of Bonn, Australia - The University of Sidney and other countries [4].

To overcome the problem of localization is the most difficult, but it can be solved with the help of Satellite Navigation System (SNS) such as GPS and GLONASS, but at the moment in the majority of cases, these systems do not provide the precision needed to control the mobile robot [5]. Furthermore, their use is limited to areas of satellite signal availability, which makes it impossible to determine the robot position within buildings and near tall buildings and trees. The most accurate solution to the navigation problem can be solved with the help of machine vision-based camera systems, laser range finders or other sensors. Such systems also allow you to generate the description of the surrounding area a robot to identify potential hazards and solve the main task. The information flow from these sensors is quite high (5 KByte / sec for the laser rangefinder and up to 8 MB / sec for video cameras), so that information processing methods require complex algorithms and significant computing power [3].

Nowadays laser sensors become popular among developers. It provides a highly accurate measurement of distance [6]. They can be used on mobile robots to produce a digital description of the map of the working area. More efficient use of the laser rangefinder sensors is possible in artificial environments, such as office and industrial environments, where the movement is performed in a plane. The laser rangefinder sensor information is sufficient to solve the problem of navigation and control of the mobile robot. The problem of controlling and navigation of mobile robot equipped with a laser rangefinder in an artificial environment is solved in this thesis.

1.2 AIMS AND OBJECTIVES

The aim of the project is the development and research of mobile robot navigation systems based on scanning laser range finder in an environment with dynamic or static obstacles.

The following objectives were set and solved to achieve the goal of the work.

- Analysis of existing methods for solving navigation tasks.
- To propose a method for controlling the movement of a mobile robot in an environment with dynamic or static obstacles
- Design a mathematical model of the system
- To provide algorithm of mobile robot via Vrep simulation environment
- Design a 3D model of a mobile robot on SolidWorks with two independently controlled wheels, ensuring its smooth dynamic movement on plane
- Development of planning of the robot path to a given point on the map.
- Carrying out experimental research on the computer model and testing in a real environment
- Make an analysis of results that was obtained

1.3 A SHORT REVIEW OF A CONTEXT

In the second chapter of the thesis was presented an analytical review of existing developments of mobile robots and an analysis of the problems which is involved in solving problems of mobile robot navigation, equipped with a laser rangefinder sensor. Considerable attention is given to the existing, including probabilistic methods for solving robot localization tasks and building the map of the working area of the mobile robot.

The third chapter is devoted to an analytical method for determining the characteristics of a mobile robot. With the help of this method was established a way of describing the surrounding world using laser rangefinder sensor and a probabilistic method of correction of accumulated errors in determining the position of a mobile robot.

The fourth chapter contains material obtained in experimental research work in which was developed navigation and control algorithms that were implemented on a computer model of the mobile robot in Vrep in combination with Matlab, as well as the mechanical part of the mobile robot was created in the center of the "Robotics" of Heriot-Watt University.

In the conclusion of the dissertation was showed the general characteristics of the research and the main conclusions of the thesis.

Chapter 2

Navigation systems

Existing navigation systems based on the GPS application and for objective reasons, they are unable to provide the required accuracy of the current position of autonomous mobile robot, while the development of local navigation based on visual feedback in combination with high-quality vision systems makes high degree of accuracy not only carrying out the determination of the robot coordinates and the surrounding objects as well [7].

Vision systems receive and process visual information about the operating environment of the mobile robot, and then transmit this information to the control system or operator. The quality of this information depends on the ability to respond rapidly to changes in the environment and success of the robot task [7].

The data obtained from the sensors are subject to further processing. Since the mobile robot must operate in transient dynamic environments, the speed of data receiving and processing should be comparable to the rate of change of the environment, as well as the rate of change of its status. In other words, the robot must process the data and make decisions in real time. On the other hand, mobile robots have limited energy and computing power, so the applied methods and algorithms have to work in low-resource settings [1].

2.1 MACHINE VISION

Vision Systems is a system consisting of the optical information device and pattern recognition system [8]. Vision systems were formed as an independent discipline at the end of the 60s. The following steps can be distinguished in the history of its development [9]:

- 1955, MIT professor Oliver Selfridge published "Pattern Recognition and modern computers." The author proposed a theoretical idea of equipping a computer with sound and image detection.
- 1960, the appearance of the first software image processing systems (mainly to remove noise from photos taken from airplanes and satellites).
- 1970, Larry Roberts, a graduate student at MIT, put forward the concept of machine constructing of three-dimensional images of objects by their analysis of two-dimensional images. At this stage, it has become more in-depth data analysis. For the recognition of objects, researchers began to use new approaches, including structural, textural and indicative.
- 1979, Professor Hans-Helmut Nagel of the University of Hamburg laid the foundations of the theory of the analysis of dynamic scenes that can recognize moving objects in the video stream.
- At the end of the 1980s, there was created robots that can evaluate the world around us and satisfactorily perform certain actions without the assistance of an operator.
- The 1990s, algorithms for image processing developed by the modular paradigm. David Marr discovered the principles of human visual perception and suggested that image analysis should be based on several consecutive levels of rising information line: from the "iconic" representation of objects (bitmap unstructured information) - to their symbolic representation (vector and attribute data in a structured form, relational structures, and so on).
- 2003, the first commercially reliable face recognition systems were released.

Nowadays, machine vision is a fully formed cybernetics section. A large variety of robots is used in different fields.

Despite the development of the area, computers still cannot "see" the same way as it does people. Cameras and other vision sensors are not equivalent to the human vision system, and while the people can rely on guesswork, machine vision systems must "see" through the study of individual pixels of the image, processing them and trying to draw conclusions using databases and pattern recognition functions.

2.2 TYPES OF NAVIGATION SYSTEMS

There are three types of navigation systems [10]:

- Global –determination of the absolute coordinates of the device when driving on long routes;
- Local – defining of device coordinates on some (usually starting) point. This scheme is in demand by developers of tactical drones and ground robots that perform the mission within a pre-known area;
- Contextual - the positioning of the robot parts of his body and the interaction with the surrounding objects, which is important for devices with manipulators.

Global methods are based on the fact that before the mobile robot movement, map of the area is completely known. Knowing their location, the finish point, and the location of all the obstacles, mobile robot using a predetermined algorithm of actions, finds the shortest path from start to finish and then overcomes this way. The most frequently used in practice, global navigation methods include wavefront methods, A* and so on [4].

The advantage of global navigation techniques is the ability to plan the optimal route based on the global information environment. The most significant drawback of such methods - increased computational complexity and the need for storage maps (often large). The need of performance of intensive calculations and storage of the map of the environment can lead to the considerable energy consumption of the navigation system of the mobile robot [10].

Local navigation methods used in those cases where mobile robot does not know the global map of the environment or obstacles in the environment have a dynamic character (can appear and disappear or change their location). In this case, mobile robot receives navigation information about the local area of the environment, within the limits of its sensors. The advantages of local navigation methods should include their computational simplicity. Disadvantages of these methods in comparison with global navigation methods are in deviation from the optimal route and a more complex procedure localization of the mobile robot in space [10].

The developed module has a local navigation system since it is assumed to use in the advance unknown environment.

Navigation systems are classified one more feature - they can be passive and active [4]. The passive navigation system includes receiving information about their coordinates and other characteristics of their movement from external sources. For example, navigation via GPS. But the use of a local GPS navigation systems is undesirable, since guaranteed accuracy in the best case, is only 2.1 meters [4]. This accuracy is not acceptable for the local navigation tasks.

Another passive concept of local navigation is using beacons. The goal is to place it in the robot action area, which is processed by an on-board microprocessor. Such system cannot be used in the non-deterministic environment, and therefore the solution is only a certain narrow range of tasks. Active navigation systems are designed for locating exclusively without using data from external sources [10].

2.3 Vision Sensors

Let's consider the technical devices that allow getting information about the surrounding area.

Laser scanner. It allows calculating the distance to the point at a long distance (4000 meters) with a processing rate of 222,000 points per second with an accuracy of 1 mm [11]. However, these scanners are not designed for ordinary consumers, because they have a very high cost. Also, there are several problems with their use in places where there is a glass (due to reflections) and underwater (due to scattering).

Ultrasonic sensors. Despite its relatively low price, sonar unpopular because of the low accuracy of measurement, the small angle and large response times [11].

The infrared range finders. Despite its cheapness and availability of their field of application, it is limited due to the low measurement accuracy and low limit range - about 150 centimetres [11].

Odometer - is a device that allows you to calculate the distance travelled by data received from the drive (for example, wheels) [4]. In real life, odometers are subject to the negative effects, such as and wheel slip. If the assessment of distance is relying solely on the odometer, the accumulated error can not only make it difficult to build the maps, but make this task impossible. Therefore, the odometer is used as an additional sensor in combination with other sensors.

Video cameras can give more information about the surrounding area than other sensors [8]. Nowadays, to solve the challenges of vision widely used the stereo system, consisting of two coupled cameras. Such a system makes it possible not only to calculate the distance to obstacles but also to build a 3D-model of the world [8]. The most common problems when using the camcorders are high computational complexity, inaccurate calibration, depending on the lighting, the erroneous data when working with reflective surfaces or the homogeneous environment.

Inertial measurement unit. The aim of the inertial navigation is to determine the acceleration of the object and its angular velocity [12]. With the help of these data, it is possible to determine the location of the object, its speed, and direction. Each inertial measurement unit includes at least an accelerometer and a gyroscope [12]. Such devices are capable of delivering a large number of parameters of movement coordinates, speed, acceleration, bearing. Because of this, inertial navigation techniques are increasingly being used to solve problems of navigation.

Navigation systems that can simultaneously measure distances in different directions, based on radar and sonar is difficult. Radar and ultrasonic sensors usually play an auxiliary role, due to insufficient accuracy. The advantage of laser scanning rangefinder is a high level of accuracy. The disadvantages are dependence on the air transparency, limit of the maximum measured distances and the lack of measurements of other characteristics (color, transparency) of the observed objects.

Due to its advantages of laser rangefinders are successfully used for equipping mobile robotic systems and even become the de facto standard equipment, which is part of the information systems of modern mobile robots.

2.4 REVIEW OF EXISTING SOLUTIONS

One of the key functions, which should have an autonomous mobile robot, is the ability to determine the parameters of its position accurately and build maps of the area. In other words, the robot must solve the problem of SLAM [5]. The basis of any method of SLAM is to measure the distance to the objects of the surrounding space and evaluate change its position about them. To measure the distance, it uses, for example, laser rangefinders, or LIDAR [13]. Using data about the distance to the object and the current position of the robot it builds a map of the surrounding space. Depending on the distance measuring devices that are used, the maps can be planar (2D) or volumetric (3D).

Nowadays, there are a large number of systems that solve the SLAM [13]. For example, Extended Kalman filter, Monte Carlo method (particle filter), SLAM on graphs. In addition to these methods, there are other, less common, but they all have their limitations and disadvantages, which again underlines the shortcomings of existing solutions and the need to conduct research in the field of SLAM. Most of these systems - non-profit or narrow-profile. Due to this, the technology has not yet reached a level to solve tasks in the field of vision accurately. However, Google has made a big step forward, demonstrating a new technology called Project Tango [15].

Project Tango - is the technology of construction of 3D-model of the surrounding space [15]. It is designed for use on special smartphones and tablets from Google. Using data from multiple cameras and depth sensors, the device can build a 3D model of the surrounding area in real time [15]. It is a revolutionary technology that has no analogs. However, the project in the development stage, and does not exist in as a final product.

A major success in the construction of autonomous robotic systems was reached by US researchers from NASA (Spirit, Opportunity, and Curiosity). Each of this mobile robot is a six-wheeled chassis (Fig. 1), with the solar panels that provide electricity to the rover, and manipulator to collect Martian soil samples and other different scientific equipment [16]. Robots designed to study the surface of Mars, which is a highly rugged mountainous terrain with craters and valleys. It is controlled by the program drawn upon the Earth based on the research plan and the current position of the robot. Programs are sequences of elementary commands for chassis, robot arm, and other equipment.

Fig. 1: NASA rover [16].

Success in this area can be seen in regular competitions of Grand Challenge, held under the auspices of the US Department of Defense DARPA [17]. The purpose of the competition is to make the movement of an autonomous vehicle by the specified route without human intervention. In 2004, out of 26 participants, the robots could not overcome predetermined distance, in 2005 five cars reached the finish line of the 230-kilometer track. Competitions in 2004 and 2005 were carried out in the desert place, and in 2007 - at the polygon, under the conditions of the city, where the participants had to take account of city traffic, road signs, and traffic lights. One of the cars participating in the competition of 2005 year, is presented in Figure 2 [17]. This mobile robot of the team at Stanford University was created on the basis of the Volkswagen car. According to published data [17], the car is equipped with servo-drives for the rotation of the steering wheel and change the mode of operation of an automatic transmission. Almost all of the robot's sensors are installed on the roof: the five laser scanning rangefinder SICK LMS291, a color video camera and radar [17]. Also, information of the robot system includes sensors satellite navigation system - GPS and inertial measurement unit. Computing information system consists of six computers based on Pentium M processor. One of which is responsible for the processing of video images, another one for recording all of the information to the hard disk. The third one for the execution of the rest of the navigation software and motion planning, and the remaining three are in reserve.

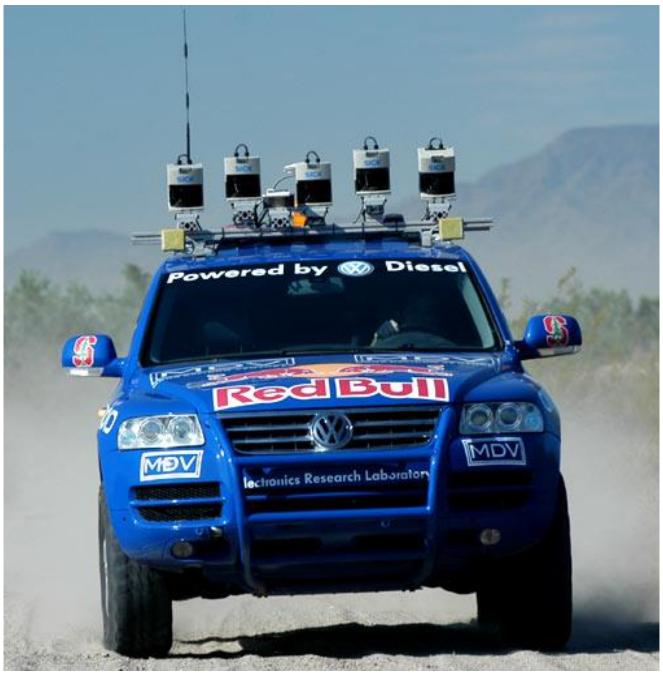


Fig. 2: Stanley mobile robot in competitions of Grand Challenge 2005 [17].

Navigation of the mobile robot during the competition was carried out at the expense of multi-sensor information from a global positioning system of data inertial measurement system and odometer sensors using a Kalman filter. This solution has allowed increasing the accuracy of determining the position of the robot compared to the GPS raw data, as well as get information about the robot's position at the moment of loss of signals from satellites.

Laser rangefinders allow generating a very accurate picture of the topography of the working area in the process of movement. It allows to identify obstacles on the path of the robot and must ensure traffic safety at high speeds.

In addition to the extreme areas of human activity, mobile robots are used in everyday life. Technology is designed for navigation and control robots that can be used in creating a robot helper for a human.

One example of the use of mobile robots under everyday life conditions is a robot guide project in 1998 for the National Museum of American History [18]. The robot is designed to conduct tours for visitors in the museum. Minerva robot (Fig. 3) is equipped with a scanning laser range finder, a video camera, and several ultrasonic sonars. The laser rangefinder is installed to measure distances in the horizontal plane. Its information is used to construct the museum plan and determines the position of the mobile robot. The camera is directed vertically upwards and also serves to localize the robot when people surround the robot, and according to the ultrasonic sensors makes it impossible.

Fig. 3: Minerva in the Smithsonian's National Museum of American History [18].

Laser sensors allow to ensure the safety of the robot movement: preventing collisions with objects that do not fall within the range of the laser rangefinder. Minerva uses two types of maps to determine his position: built according to the plan of the museum rangefinder and the resulting images on a camcorder. Measurements of the first type are used for the construction plan and the determining of the own position, and the second for filming people and find a safe path to the target point.

At the moment, there is no universal solution to the problem of SLAM, using combinations of the above sensors and software for these sensors. Due to these sensors do not always provide accurate measurements and are subject to the accumulation of errors over time, evaluation of the robot position could differ from its actual position. To overcome this limitation, most algorithms that implement the SLAM, use the following approach: the extended Kalman filter, particle filter or SLAM graph. Let's consider these in more detail.

2.5 KALMAN FILTERING METHOD

Kalman filtering method [19,20], is intended to produce assessments of the state and the statistical characteristics of the system by some measurements, suggested by R. Kalman in 1960. In solving the localization problem under consideration of the Kalman filter is used to obtain an estimate of the position of the robot at the same time evaluating the situation of some beacons [19]. The beacons appear as features of the outline, which can be isolated from the image (Fig. 4). We can say that this method is an extension method of navigating the local terrain features discussed above. In practical implementations of this method, the number of beacons ranges from one to several hundred. In some cases, the number of beacons is unknown a priori but is recognized and dynamically determined by a robot [20].

The successful operation of this method is determined by the ability to distinguish and uniquely identify the beacon.

If we assume that the position of the robot in the plane is described by three variables (two coordinates and orientation angle), and the position of each beacon two (coordinates), then to k beacon of the Kalman filter should evaluate both the state vector of 2k + 3 variables. Some beacons determine the size and complexity of computing the filter (which increases in proportion to the square of the dimension of the state vector).

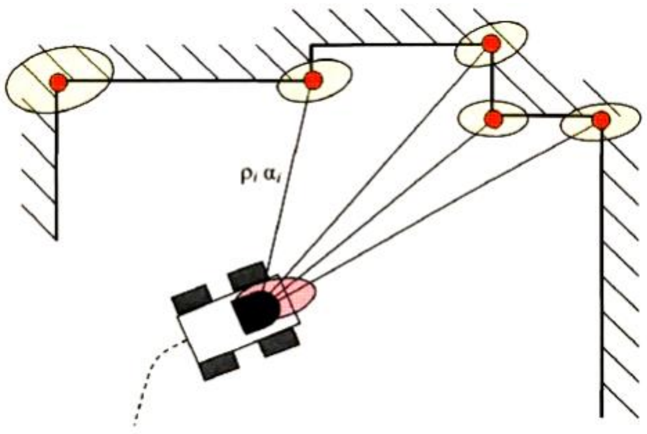


Fig. 4: Application of Kalman filters for navigation. Ellipses show the probability distribution corresponding to the position of beacons and the robot [20].

In the original version of the Kalman filter model of perception and the model change of state is considered as linear with normally distributed noise. There is extensions filter, taking into account the non-linearity due to the linearization. The main advantage of the Kalman filter is that it estimates the full probability distribution of the robot and the position of beacons by the measurement accuracy. In simplified form, the filter equations are formulated as follows [19,20]:

$$\boldsymbol{p(x\,|u,x^i) = Ax^i + Bu + \varepsilon_{control}}$$

$$\boldsymbol{p(z|x) = Cx + \varepsilon_{measure}}$$

where the x - state vector, the z - vector measurements, and u - the control vector, A, B, C - matrix system, $\boldsymbol{\varepsilon_{control}}$ and $\boldsymbol{\varepsilon_{measure}}$ is characterized by the uncertainty of the robot motion and measurement of noise respectively. [19] states that the standard Kalman filter equations can be obtained from equations Bayesian filter.

2.6 THE METHOD OF PARTICLE FILTER

The method of filtering particle (particle filter) can be considered as an approximation and extension of the Kalman filtering method to solve the problem of localization and building maps of the mobile robotics [21].

In this method, as in the Kalman filter, made an attempt to estimate the probability density function of the robot position and state of the world in the form of $\boldsymbol{p}(\boldsymbol{x_t}, \boldsymbol{m}|\boldsymbol{z^t}, \boldsymbol{u^t})$, where $\boldsymbol{x_t}$ is the position of the robot at time t, m – is the map, $\boldsymbol{z^t}$ is measurements collected during robot movement 0...t, $\boldsymbol{u^t}$ is the data on the movement of the robot (or odometer on the management of information).

If the Kalman filter probability distribution is represented as a normal variable parameter, in this method, the probability distribution is considered as some hypotheses. The method uses the following formula to calculate the density of distribution:

$$\boldsymbol{p}(\boldsymbol{x_t}, \boldsymbol{m}|\boldsymbol{z^t}, \boldsymbol{u^t}) = \boldsymbol{p}(\boldsymbol{m}|\boldsymbol{x_t}, \boldsymbol{z^t}) * \boldsymbol{p}(\boldsymbol{x_t}|\boldsymbol{z^t}, \boldsymbol{u^t})$$

Which allow assessing the trajectory of the robot, and then build a map based on the received path. The distribution $\boldsymbol{p}(\boldsymbol{m}|\boldsymbol{x_t}, \boldsymbol{z^t})$ is corresponding to the representation of the map can be obtained analytically if the trajectory of the robot is known, consisting of a sequence of positions $\boldsymbol{s_t} = \boldsymbol{x^t}$, and the measurement $\boldsymbol{z^t}$ respectively relating to each position of the robot. Evaluation of the second distribution $\boldsymbol{p}(\boldsymbol{x_t}|\boldsymbol{z^t}, \boldsymbol{u^t})$, represents the assumption that the trajectory of the robot takes place using a particle filter.

The algorithm of particle filter estimates the number of hypotheses about the mobile robot position at the same time. Each hypothesis can be represented as point particles in a plane which characterizes the position of the robot projection $\boldsymbol{x_t^{(i)}}$, that gave the name of the filter. With the hypothesis related probable trajectory of the $\boldsymbol{x_t^{(i)}}$ ... $\boldsymbol{x_t^{(i)}}$and grip map of the area $\boldsymbol{m_t^{(i)}}$ which was built on the basis of assumptions about the trajectory and received measurements. The set of hypotheses particles, each of which is characterized by its own weight, is a full probability distribution of the position of the robot and the map $\boldsymbol{p}(\boldsymbol{x}_{\boldsymbol{t}}, \boldsymbol{m}|\boldsymbol{z}^{\boldsymbol{t}}, \boldsymbol{u}^{\boldsymbol{t}})$

The algorithm consists of four iterative steps [21]:

- Sampling. Generating the new set of particles $\{\boldsymbol{x}_{\boldsymbol{t}}^{(\boldsymbol{i})}\}$ of the current distribution $\{\boldsymbol{x}_{\boldsymbol{t-1}}^{(\boldsymbol{i})}\}$ on the basis of the probabilistic model of the motion to the distribution of $\boldsymbol{\pi}$. That is, for each particle the hypothesis of $\{\boldsymbol{x}_{\boldsymbol{t-1}}^{(\boldsymbol{i})}\}$, occurs generation of new hypotheses concerning the robot position after the selected period of time (Fig. 5 (a)). Assumptions are made on the basis of available data on the movement of the robot, it can be odometer data or the results to determine the relative movement of the robot based on a laser range finder results. Distribution of new hypotheses is usually formed as normal on the basis of knowledge of the accuracy of the data used (the accuracy of odometer readings of sensors or accuracy of the solution of local problems of navigation).
- Importance Weighting. Assigning a new weight with $\{\boldsymbol{w}_{\boldsymbol{t}}^{(\boldsymbol{i})}\}$ each newly generated particle based on the measurement mode $p(z_t|m_{t-1}^{(i)}, x_t^{(i)})$ on the map, which is associated with each particle, as well as Bayes' rule. The new value of the weighting factor depends on how the new hypothesis corresponds to the existing position of the robot map: the best readings of rangefinder z_t of robot position $x_t^{(i)}$ coincide with the map of $m_{t-1}^{(i)}$ As a measurement model it uses estimated probability scan acquisition on this map with the help of ray tracing.
- Resampling. Revision of new hypotheses and the normalization of the weights of the particles. Based on the calculated weights screening of particles occurs so that it remains only in consideration of certain, finite number of hypotheses with the highest weights. The weights of the remaining particles are normalized so that the sum of the weights is equal to one. This step eliminated the unlikely hypothesis of the robot position.
- Map estimation. Evaluation of map. For each particle based on the latest measurements z_t and assumptions about the robot's position $x_t^{(i)}$ and being completed in the map of the area $m_t^{(i)}$.

These steps are performed recursively through given time interval each time with new measurements of the range finder.

Consideration of several hypotheses allows compensating for errors that accumulate in the relative decision of the local navigation tasks. At the same time, the more particles are considered, the better the algorithm to cope with accumulating errors, but the higher requirements for compute-intensive and memory to store maps associated with each particle. A very important step is the generation of new hypotheses. Their distribution should include the correct hypothesis robot position, but at the same time increasing the number of assumptions also leads to an increase in computational costs for the third and fourth steps of the algorithm.

The generation of new hypotheses by determining the position of the robot local navigation techniques can significantly reduce the number of proposed particle generation compared with similar data based on the odometer. The disadvantage of this approach is that existing local navigation algorithms discussed above, do not provide the accuracy of estimates of determining the position of the robot itself, so to generate new hypotheses using a normal distribution with parameters obtained by statistical data on the accuracy of the algorithms. This distribution does not depend on the nature of the mobile robot or environment and can lead to the generation of unduly a large number of particles or omission of true hypotheses from consideration when the precision of navigation task falls, for example, when driving along a wall or a corridor. Figure 5(b) shows one iteration of the algorithm: the proposal of new hypotheses based on the model of the motion of particles and screening by the probability distribution model.

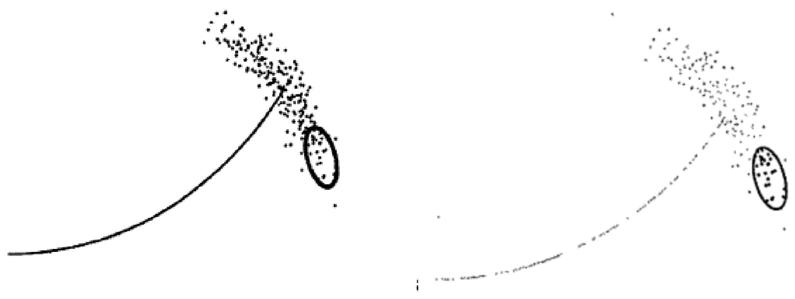


Fig. 5: The working principle of the particle filter. The proposal of new hypotheses (a). Screenings of hypotheses by measurements (b) [21].

2.7 Monte Carlo Localization for Mobile Robots

The Monte Carlo localization is a special case application of the particle filter for the localization of the mobile robot at a certain working area map [22]. The algorithm is also iterative and represents the robot position in the form of probability by a certain number hypothesis - particles. Map of the area in which the robot moves is considered as a given and does not change; it is used to build the measurement model, the second step of determining the weights of the particles. Accordingly, the fourth step of the particle filter - building map is not needed. The initial number of particles and their position is determined by the uncertainty of the initial position of the robot. When the position of the robot is not known, the particles will be distributed throughout the map. Work of particle filter reduces the number of hypotheses and to clarify the position of the robot. Fig. 6 shows the distribution of particles at different stages of the Monte Carlo algorithm. At the beginning position of the robot is unknown, and the particles are distributed all over the map, in the following figure, after the robot has made some movement and measure, there was a screening of the particles with small weights, the robot position uncertainty has decreased. Because the map has a certain symmetry particles grouped in two similar areas on the map.

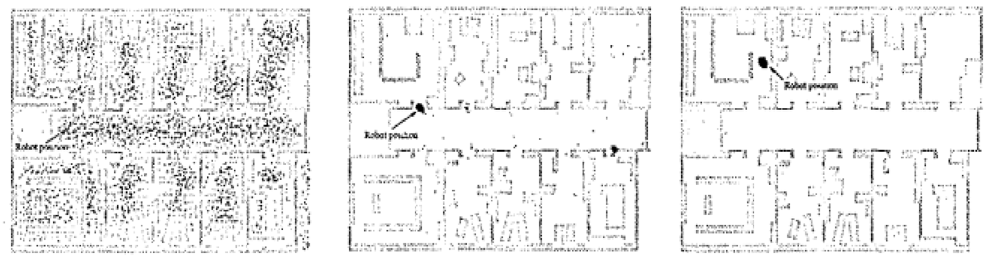


Fig. 6: Global localization of the mobile robot Monte Carlo [22].

It should be noted that the successful operation of the algorithm is weakly dependent "on the accuracy of the map as a probabilistic approach allows us to cope with the disparity map the world around them by considering several hypotheses about the mobile robot position. A localization algorithm using a particle filter and hence the Monte Carlo model is movement by which the appearance of new particles. If this model is inaccurate, for example, in the case of using odometer, it increases the number of the particles, and hence the complexity of the algorithm. If the model is based on the results of one of the methods for solving local navigation problem, in this case, the number of particles is less, but there is a possibility to miss certain hypothesis of mobile robot position from consideration, since the existing methods of local navigation do not define a complete probability distribution of the mobile robot position, then such information is taken from the statistical data on selected local navigation algorithm, as in the case of the filter particles.

2.8 SUMMARY OF THE SECOND CHAPTER

The first chapter describes the information systems of modern mobile robots and identifies the benefits of using different navigation methods.

Identify the problems that arise when navigating the mobile robot equipped with a laser scanning rangefinder in an artificial environment. The main challenges are to determine their position in the robot's environment and to build an understanding of the world around, which is Localization and Mapping(SLAM).

Local navigation robot has a significant drawback - eventually accumulating error, so to ensure the operation of the robot is necessary to solve the problem of global navigation. This problem can be solved by using probabilistic methods to examine several hypotheses about the robot's position and if possible to compensate for accumulating error. The above methods of use of the global navigation local navigation solution algorithms to generate new hypotheses. In this case, the main disadvantage of existing methods of local navigation - the lack of estimates of the accuracy of the solution - could lead to a loss of productivity and the accuracy of methods for global navigation. Necessary to develop methods for local navigation provides an estimation of the accuracy of determining the position of the robot, depending on the nature of the terrain and the mobile robot. Such methods can be used to solve the problem of global navigation with higher efficiency and accuracy

Chapter 3

Path planning

The chapter considers the problem of mobile robot control with the main element of which is laser rangefinder sensor.

3.1 CONTROLLING MOBILE ROBOT USING LASER RANGE FINDER SENSOR

«Under control of the mobile robot» means a change of motion parameters to perform a specific task: to ensure moving to the specified point, performing a predetermined path, etc. For a mobile robot with two symmetrically arranged drive wheels, the task of controlling is reduced to the task of controlling speeds of wheels on each side. Their values determine the movement of the mobile robot in the absence of wheel slippage by the formulas [23]:

$$\begin{cases} \vartheta = \dfrac{R}{2}(w_r + w_l) \\ \omega = w_{r-}\, w_l \end{cases}$$

where $\boldsymbol{w_r}$ and $\boldsymbol{w_l}$ the angular velocities of right and left wheels of the robot, **R** - radius of the wheels of the robot, **w** and, **v** - respectively angular and linear velocity of the robot excluding wheel slippage.

Drives of modern mobile robots contain an internal control loop, allowing keep the desired angular speed of the wheels. However, to control the changes of the position of the mobile robot only by solving the inverse kinematic problem by determining wheels $\boldsymbol{w_r}$ and $\boldsymbol{w_l}$, based on the desired **w** and $\boldsymbol{v}$, by the formula that above makes it impossible. Even in the absence of wheel slippage, the dynamics of their drives can't be identical, leading to errors. For example, if we want the robot driving in a straight line, when its initial direction is given and, both of wheels have equal rotation speed, the robot is likely to deviate from the path. The error will happen because of a difference of rotation of mobile robot wheels.

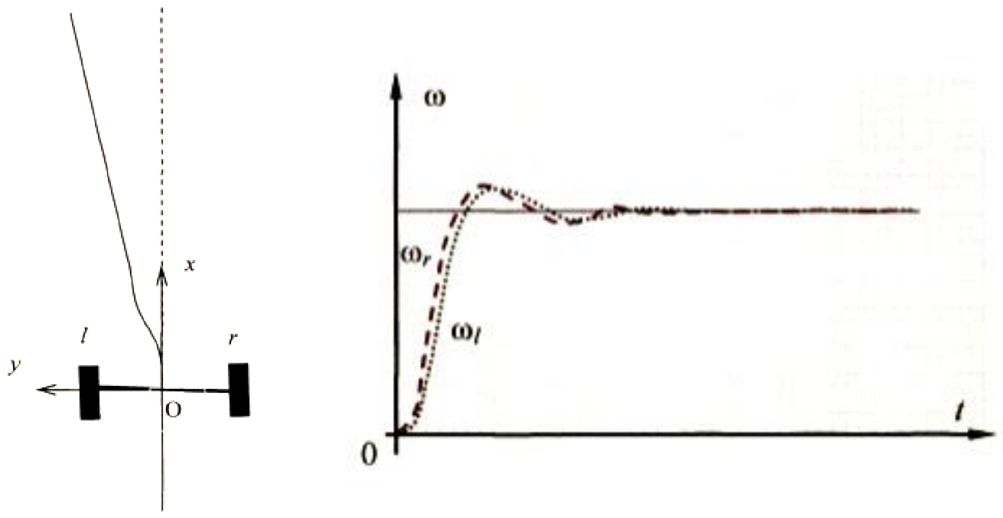


Fig. 7: The deviation of the mobile robot by a predetermined PATH (a) caused by different dynamics of wheels (b).

Therefore, the mobile robot control system should generate an additional control circuit, allowing the wheels to set the speed based on the sensor information constantly correcting the wheel velocity in order to provide movement along the desired trajectory.

As already noted, the information about the outside world, the mobile robot receives from the laser scanning rangefinder. Therefore, mobile robot control methods can be divided into two classes: methods of controlling a mobile robot from current information, and methods that take into account the information obtained in previous times. The first class is simple to implement, is not demanding on system computing resources, but at the same time, allow us to solve these locomotion tasks, such as motion along the wall, the movement along the corridor, moving by avoiding obstacles. Typically, these control algorithms are used at the lower level, by providing the operational control of the mobile robot. The working principle consists in the continuous running of the cycle: getting data from sensors - the calculation of new parameters of motion, which were immediately accepted for execution. The methods of the second class are used to control the outside world, formed according to the laser rangefinder sensor, and information about the current position of the robot obtained as a result of navigation. These methods are much more complicated regarding algorithmic and require large computational resources, but they are much more resistant to the various forms of a cross-section of the relief and have a wide application, in particular, such methods are widely used in [23,24].

3.2 PROVIDING MOVEMENT ALONG THE WALL

Usually, the mobile robot can be guided on the wall located on its one side (right or left), which allows the robot to ensure the return to the starting point on the same wall when changing orienting side [25]. Firstly, the robot moves, for example, along the right wall, then at any moment it can turn around and start moving in the direction of the starting point along that the walls, but located to the left of it. This problem can be solved by using a robot control algorithm of the first class.

The problem is formulated as follows. The mobile robot has to move at a predetermined distance from the object which is a wall. Wall is considered to be inseparable for the robot that does not contain gaps that exceed the overall size of the robot. The mobile robot model was previously considered which can be changed by the speed of translation and rotation by changing the speeds of the drive wheels. The linear velocity, is considered a given and constant, and robot motion control is carried out by changing the rotational speed. The robot is equipped with the laser rangefinder, scanning the surrounding area in the plane of movement of the robot. Information from the sensor represented as an array of distances that is used to control the robot [27].

Let's consider the simplest case, when the wall is perfectly flat. In this case, the robot should move in a straight line positioned at a predetermined distance from the wall. In the working area of the robot, there are any no other obstacles except the wall.

Figure 8 shows the position of the robot and Figure 9 shows the data that gives the laser rangefinder sensor.

Where, **XOY** - coordinate system of the mobile robot, coinciding with the coordinate system of the laser rangefinder sensor, **d** - a predetermined distance from the wall to the desired trajectory of the mobile robot, **z** - current distance from the mobile robot to the wall, $\boldsymbol{\varphi}$- is the angle between the wall and **OX** axis of mobile robot.

In this case, we can determine the equation of a straight-wall at two points from an array of range measurements related to the wall. For the ith point:

$$x_i = p_i \cos a_i$$

$$y = p_i \sin a_i$$

then the equation of the wall can be written as an equation of a line passing through two adjacent points [25]:

$$(x_i - x_{i+1})y - (y_i - y_{i+1})x = x_i y_{i+1} - y_i x_{i+1}$$

The angle between the wall and the x-axis:

$$\varphi = \tan^{-1}(\frac{y_i - y_{i+1}}{x_i - x_{i+1}})$$

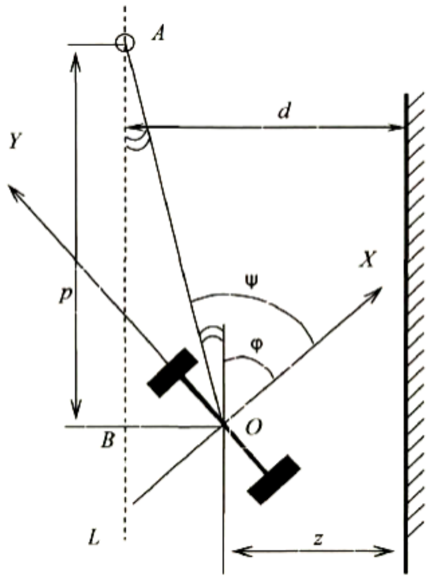


Fig. 8: Model of Mobile Robot

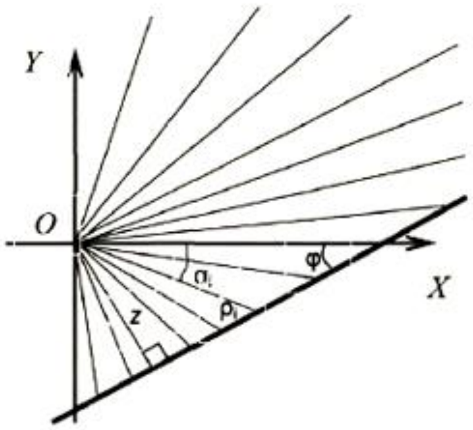


Fig. 9: Data from laser sensor

The distance to the wall [25]:

$$z = (x_i y_{i+1} - y_i x_{i+1}) / \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1)}^2}$$

We need to create a control that takes the mobile robot center - the point **O** to the line **L.** For this purpose, in each quantum of time it will produce a robot control in order to achieve a virtual beacon which is located at the desired trajectory at a predetermined distance **p** in front of the robot - point **A** in Figure 9. Then the angle between the direction of the **X**-axis robot and the direction of the beacon from the triangle **OAB** is $\psi = \frac{\pi}{2} - \tan^{-1}\frac{p}{d-z} + \varphi$

The value of this angle will be the error control system. To turn the robot to the beacon, we will take its angular velocity proportional to the angle $\boldsymbol{\varphi}$

$$w = k\psi$$

where **k** – coefficient parameter of control.

Let's consider the general case where the horizontal section of the wall - arbitrary smooth curve (Figure 10).

In this case, we assume that the virtual beacon - point **A** - is on the line, placed at a predetermined distance **d**, from the tangent to the closest point of the outline. The tangent equation to the next point of an outline in relation to the robot can be considered as equivalent to the equation of the straight line passing through the closest point next to her.

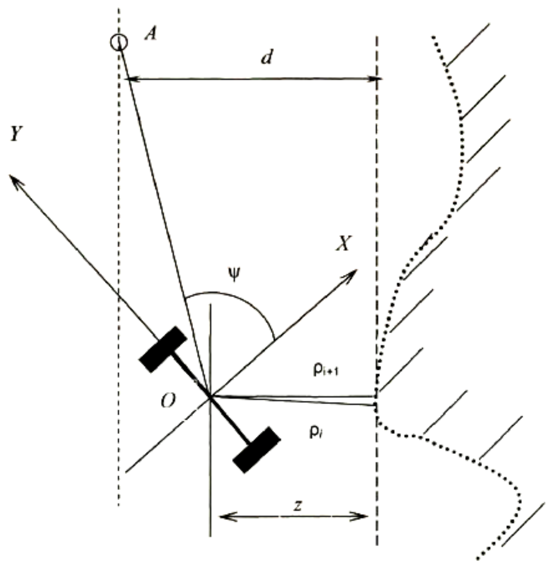


Fig. 10: Movement of mobile robot along a non-ideal wall movement

It is similar to the first case where we determine the equation of the straight line, and then angle $\boldsymbol{\psi}$ and angular velocity $\boldsymbol{\omega}$.

However, in the room where there are lots of furniture of various sizes, wall section is non-smooth, and irregularities that were formed by the clash of facilities in the room make greatly affect the character of the movement of the mobile robot. Data from laser range finder sensor is used to construct a tangent line that determines the direction of movement of the mobile robot, the points of discontinuity of its inclination and hence the position of the beacon can vary dramatically, causing jerks and even loss of robot vision. Loss of vision can also occur in a situation where the wall for some reason disappears from the field of view of the robot, such as when traveling along walls with sharp corners.

Possible options for improving the algorithm is to create a movement to the beacon, set a line parallel to the chord, which is conducted through the nearest to the center of the coordinate point of a range finder with the number **I**, and the last point of the scan associated with it. That is a point to the number **k(k> i)**, for all the points with the number **p** such that **I <p <k**, the distance between **p** and **p + 1** does not exceed a predetermined value f (Fig. 11). In this case, there is no operation of constructing a tangent to the cross section of the relief, the point at which the chord is built, far apart and therefore the angle of its direction does not change a dramatically as the angle of the tangent in the previous case.

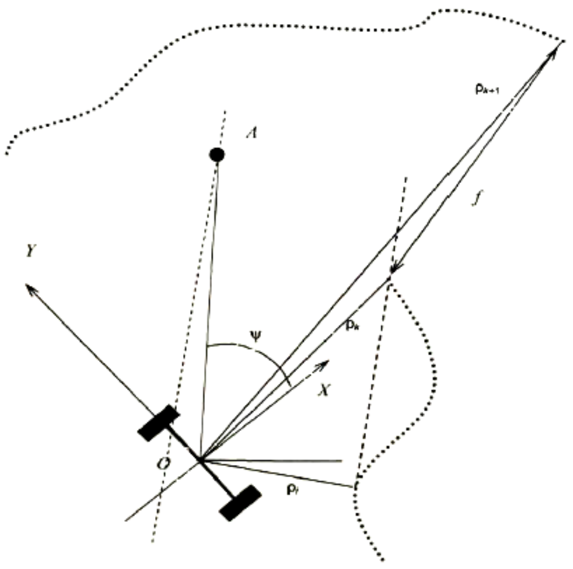


Fig. 11: An improved algorithm for motion along the wall

Such a modification of the algorithm can significantly reduce the influence of the wall roughness, the trajectory of movement of the robot becomes smoother.

When moving along the wall at a certain distance from it, the average length of beams of the sensor remains constant. If it increases - thus, the robot moves away from the wall, and if it decreases - close to the wall. This fact can be used to simplify the method of motion along the wall.

Let's introduce the concept of a "**virtual probe**" as the distance from the cross section of the relief in a certain direction on the range finder coordinate system. This can be used as a measurement of deviation angle from the axis that is equal to -45 °. During the movement along the right wall (Figure 11): if the wall is going to the left- the length is increased, it is necessary to turn to the right if the wall goes to the right, the probe becomes shorter, and the robot has to turn to the left. As the control of error can be used a difference between the reference probe length $\mathbf{p}$ and the current measurement $\boldsymbol{p}_{-45}$ of rangefinder in the selected direction: $\boldsymbol{w} = \boldsymbol{k}(\boldsymbol{p}_{-45} - \boldsymbol{p})$.

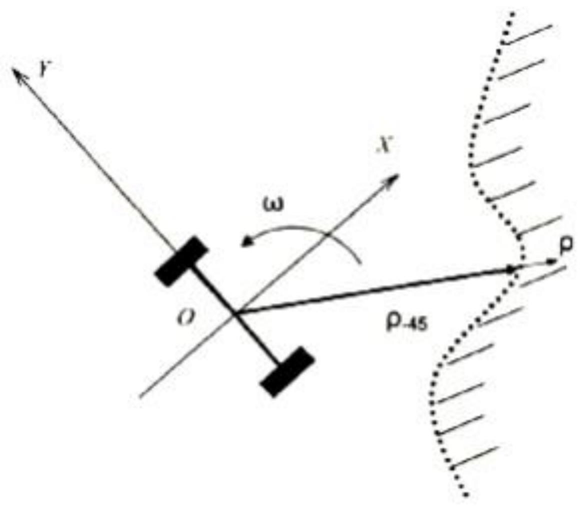


Fig. 12: Movement along the wall with the help of the virtual probe.

k and **p** are the parameters of the algorithm responsible for the dynamic turning of robot trajectory and its proximity to a wall, respectively.

This method also allows you to resolve the problem of the passage of a sharp turn to the right wall. In this case, the beam probe length will increase dramatically, and the robot will be forced to turn to the right to follow the wall.

It should be noted that by using the methods that were considered require further analysis of the image to detect obstacles in the robot's path. In the case of detecting any object in the front field of view, the robot speed must be reduced, and additional maneuvers should be taken to avoid obstacles, and then the movement along the wall can be continued.

3.3 MOVEMENT TO A GIVEN POINT ON THE MAP

The image that was formed by laser rangefinder sensor contains limited information about the world. Therefore, it is easy to imagine such a cross-section of relief on which the methods suggested above cannot solve the problem following along the wall. Also, with the use of a mobile robot control method according to the current scan, basically not possible to solve the problem of motion to a given point on the map and returning to the starting position, because the robot has no information about his situation and sufficient description of the world. To solve these problems, it is necessary to apply the methods of the second type of control - methods in which the robot uses a representation of the outside world and own position in it [4,25,26].

3.3.1 Path planning on the map

Using information about its own position of the mobile robot on the map, obtained by solving the localization tasks and the method that was described in Chapter 2, allows to set the task of moving the mobile robot to the target point at specific coordinates in the working area, which should be reached with a certain accuracy. This method can be used for solving the problem discussed above movement along the wall: on each step of the algorithm there can be an effective objective - the point remote from a wall at the set distance. The movement directed to that point will be equivalent to the movement along a wall.

Maps for solving navigation tasks allows searching for the path to the specified point with obstacles on the map. There are many well-known methods, which allow finding the way to a given point on the grip map. These are such algorithms as wave algorithm, Dijkstra algorithm, A * algorithm, and others [4]. The received trajectory of the movement can meet various criteria of optimality.

Using these algorithms for path planning robot equipped with a laser rangefinder, to the point, it is necessary to take into account the following features [26].

- Firstly, any changes on the map in case of appearing of new objects in the zone of movement of the mobile robot and in the case of changing the robot coordinate.
- Second, inaccurate determination of the position of the robot and an increased risk of collision with obstacles.

Any changes on the map, measurement inaccuracy of the laser range finder, a navigation error in path planning and changes in the working area of the relief can occur at each step of the robot localization algorithm that was proposed in chapter 2. Accordingly, the route of the robot can be redesigned with a new starting position. However, in the case where there are two alternative routes of travel to the target point, frequent rescheduling can lead to a situation where after the beginning of the movement in one selected route, the second way can be preferred [4].

The time that was spent on planning the route depends on the size of the map and distances between locations. In general, this time is non-deterministic and execution path search algorithm in real time is difficult.

To solve these problems, in the thesis was used and developed a well-known algorithm for finding the path to the target point [15], [4] and control the movement of the robot.

3.3.2 Modification of search algorithm of the mobile robot.

As path search algorithm to a point on the map has been used wave algorithm to find the shortest route between two points on the map. Every cell of the map corresponds to a busy or free for movement point of the plane. Wave algorithm allows you to find a path between two free cell of the map [28]. The sizes of the robot are considered due to increasing the number of obstacles and every cell corresponding to the points on the plane, which is located near the obstacles at a distance not exceeding the size of a mobile robot, also marked as busy. Expansion of this algorithm consists of the following steps.

Once the route to a given point is found, the reference points of the route can be determined. These are the center points of successive cells, each of which is in the line of sight of the positions corresponding to neighbouring points. The condition of direct visibility means that the straight line drawn between consistently, chosen route points doesn't cross the map cells marked as an obstacle.

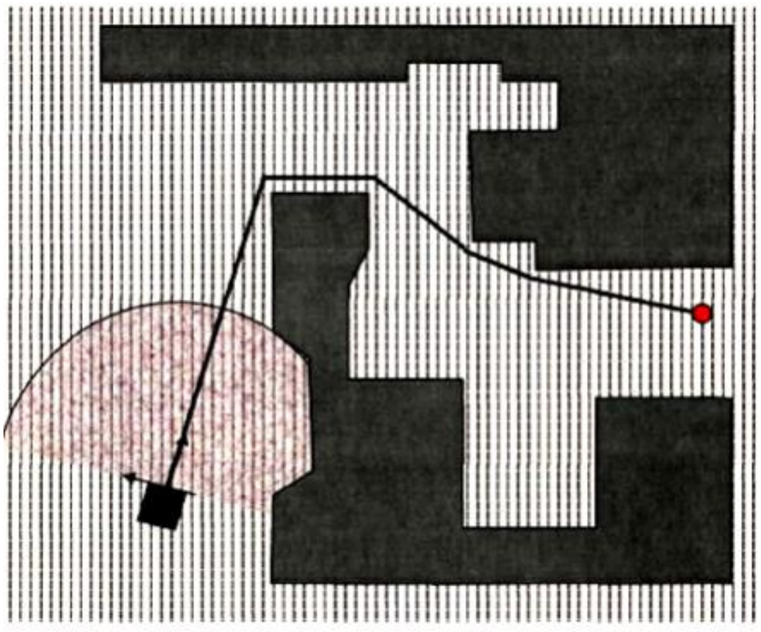


Fig. 13: Path planning using wave algorithm

To ensure in traffic safety in the environment with obstacles, it is necessary to adjust the route so that it goes as far as possible from occupied(busy) areas. A penalty can do the correction for its proximity to an obstacle. This map can be obtained from the original grid map using the algorithm thinning free zones. This algorithm is known from the theory of the image processing [11] and is used for thinning the binary image. The algorithm is executed recursively, at each step, there is thinning of the image represented by the free area maps. As a result of this algorithm, we obtain a modified map of the area, with a clear way cell which is associated the value of the penalty is inversely proportional to the distance to an obstacle. Schematically working principle of the thinning algorithm is shown in the following figure 14. The color of map section characterizes the value of the associated penalty: if the map section is becoming darker, the penalty will be increased.

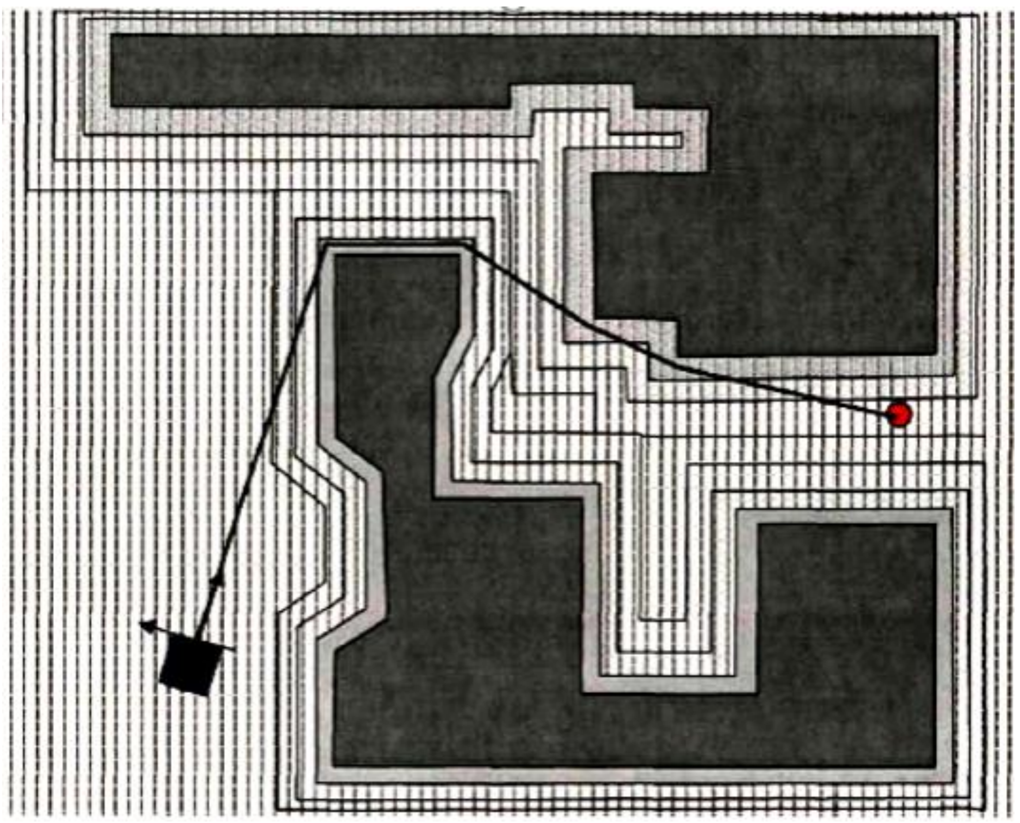


Fig. 14: Construction of map with a thinning of free areas.

To make the robot route safer, you need to "**shift**" it in an area remote from obstacles. This operation can be carried out according to the penalties received by the map: the reference point of the route is shifted so that the penalty was minimal. In order to "**shift**" occurs along the entire length, it is necessary to add a set of reference points so that the distance between the adjacent dots does not exceed a predetermined value which should be lower than rangefinder distance.

The offset of point occurs iteratively: alternately for each point except the initial and final, as long as such offset is possible for at least one point.

Reference points can be used as serial target points, reaching it the mobile robot moves slowly toward the ultimate goal, which is the last reference point. Figure 14 shows an example of a planned route and selected reference points.

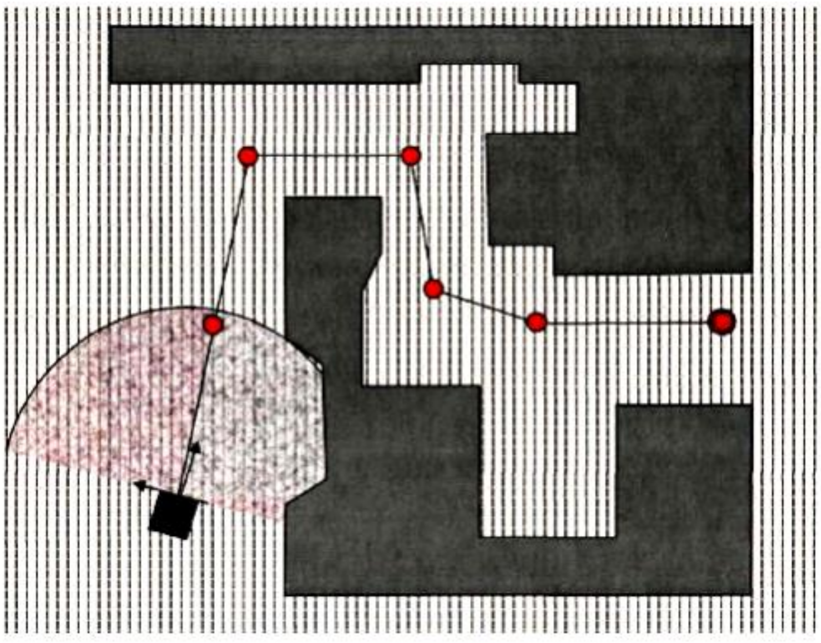


Fig. 15: The reference points of the route.

Planning for the full route at a given point and taking reference points occurs once - at the beginning of the algorithm. After that, the robot must reach a series of reference points.

Finding the way to the nearest reference point, which is currently in the local purpose of the mobile robot may be carried out by standard methods for searching the route, such as **wave algorithm**. It may also take into account penalty in order to ensure safe movement. The size of the map is limited to a range of laser rangefinder sensor. Path planning of the route to the nearest reference point can be made at each step of the recursive filtering algorithm to update the real-time map. In this case, the reference point in the path can be dynamically changed with the appearance of different objects in the path of the robot.

During the movement of the mobile robot to the next reference point may be a situation in which this target point is not free, as a result of changing the position of the working area objects or as a result of inaccuracies in solving the problem of localization and hence the imprecise construction of the map. In this case, it must be redesigned the entire mobile robot route to the final destination point in finding a freeway and the choice of new reference points

3.3.3 Movement and Trajectory

When the desirable trajectory of the movement was constructed, it is necessary to develop a drive control signals to perform a movement of the mobile robot [26].

In the thesis was considered a two-wheeled mobile robot, which is controlled by specifying the linear and angular velocity, which can be converted into the speed of the left and right wheels. The error can be measured as the "**angle of misalignment**", which is the angle between the current angular position of the robot and the direction of the trajectory of the target $\boldsymbol{\psi}$ (fig.16). If we assume that the linear velocity is constant, the robot will be managed through the angular speed reference with:

$$\boldsymbol{w = k\psi}$$

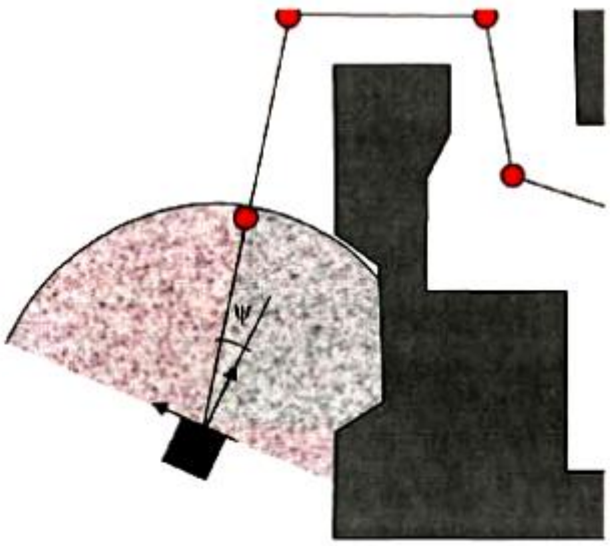


Fig. 16: Angle of Misalignment.

In the case of large mismatches of angle $\boldsymbol{\psi}$**,** the speed of the mobile robot can be reduced to perform the accurate turn. Upon reaching the robot each reference point, the target point becomes the next point in the route, and the mobile robot, without stopping, continues to move [4].

3.4 Energy-Efficient Motion Control of Mobile Robots

Another issue is an energy limitation [3]. It is one of the most important challenges for mobile robots. Mobile robots have limited energy and computing power, so the applied methods and algorithms have to work in low-resource conditions. The research shows that motion of mobile robot consumes a lot of power. Therefore, it is important to consider the energy-efficient designs and make a mobile robot that consumes less energy.

In this paper, we propose a method for navigation mobile robot, based on a combination of approaches, for both global and local for navigation methods that can reduce the cost and increase the energy efficiency of mobile robots. The proposed method involves performing navigation based on:

- information about the local environment within the laser sensor range;
- the two components of the global navigation information: the direction of the target and the distance to the target.

The proposed method has the following limitations:

- The area of which moves mobile robot consists of two types of areas: open areas and area with dynamic and static obstacles;
- processed only convex obstacles with right angles (such obstacles, for example, buildings);
- The mobile robot could not directly overcome the obstacle. If an obstacle is in the way of robot, he should avoid it;

The proposed method, in contrast to the practical implementation of known methods for global navigation, does not require storage for maps and perform sophisticated route calculation procedures, as well as route recalculation procedure for dynamic change of landscape. Thus, the proposed method is potentially less energy consuming than the global navigation methods because of the lack of need to store maps, route calculation, and recalculation. In contrast to known methods of remote local navigation, it does not require the presence of two or three permanent sensors (ultrasound, laser), and involves the use of an only one laser rangefinder sensor, which has a big range of sensing the obstacles. Since the ultrasonic and laser sensors continually generate and receive appropriate signals, they are more energy-consuming than other vision sensors. Furthermore, they are themselves more expensive devices, especially laser sensors. Due to such reasons, the proposed method minimizes the number of sensors to one.

Unlike local navigation methods proposed method loss problems and the need to adapt the sensor to the surface relief obstacles. Thus, the proposed method can be considered effective navigation method (which is critical for the autonomous mobile robots) and cost position sensor system.

3.5 SUMMARY OF THE THIRD CHAPTER

In the third chapter was considered some of the mobile robot control methods, equipped with a laser rangefinder sensor.

The existing control methods are ineffective, although they can be used to organize a robot control in simple cases, for example, when moving along the wall.

By knowing the current position of the robot and the surrounding map that constructed according to a laser rangefinder sensor, allow you to define the target point of the movement, to find on the map a clear path to it and control the movement of the robot to reach the goal.

In this chapter, a modification of the method of path planning has been proposed, which guarantees the safe movement and reaching the target point in an environment with obstacles.

Chapter 4

Modeling and experimental studies

The fourth chapter presents the results of experimental studies. They included the following steps:

- Development of mobile robot control system, which used algorithms developed in the dissertation
- Design a 3D model of a mobile robot on SolidWorks with two independently controlled wheels, ensuring its smooth dynamic movement on plane
- Import 3D model to Vrep
- Testing of control system in the Vrep modeling environment
- Make an analysis of results that was obtained

4.1 DEVELOPMENT OF MOBILE ROBOT CONTROL SYSTEM

The goal was to create a system of automatic control of a mobile robot equipped with a laser rangefinder.

The control system must perform the following functions:

- control of the mobile robot (along the wall, at the point specified by the operator on the terms of the working area)
- Determining the position of the robot
- Providing mapping of the mobile robot

Providing safety of the mobile robot: prevention of collision of the robot with obstacles and capsizing of the chassis

4.2 CONTROL SYSTEM ARCHITECTURE

Mobile robot control system is organized as a distributed system, where each module performs a specific function. Such construction allows providing the following properties of system

- the independence of individual modules
- developing and debugging of each module individually

The following modules have been developed:

- **User interface module**. Responsible for the visualization of the state of the mobile robot, display sensors and user interaction information.
- **Drive control module**. It converts ADC signals to the drive chassis, based on the current angular and linear velocity.
- **Interaction module**. Responsible for obtaining images and configuration parameters from the laser sensor via the serial interface.
- **Position module**. Implements developed in the thesis an algorithm for determining the angular and linear velocity of the robot and their integration with the help of the Kalman filter to obtain the position of the mobile robot.
- **Map module**. Implements the construction of map and responsible for the robot position correction algorithm using "particle filter".
- **Wall module**. It provides control of mobile robot movement along the wall.
- **Point module**. It provides a movement of the robot at a given point.
- **Security module**. Controls robot vision to prevent a possible collision with the obstacles
- **Event Logging Module.** Responsible for recording special diagnostic messages and operator actions to a common file for later viewing and analysis.

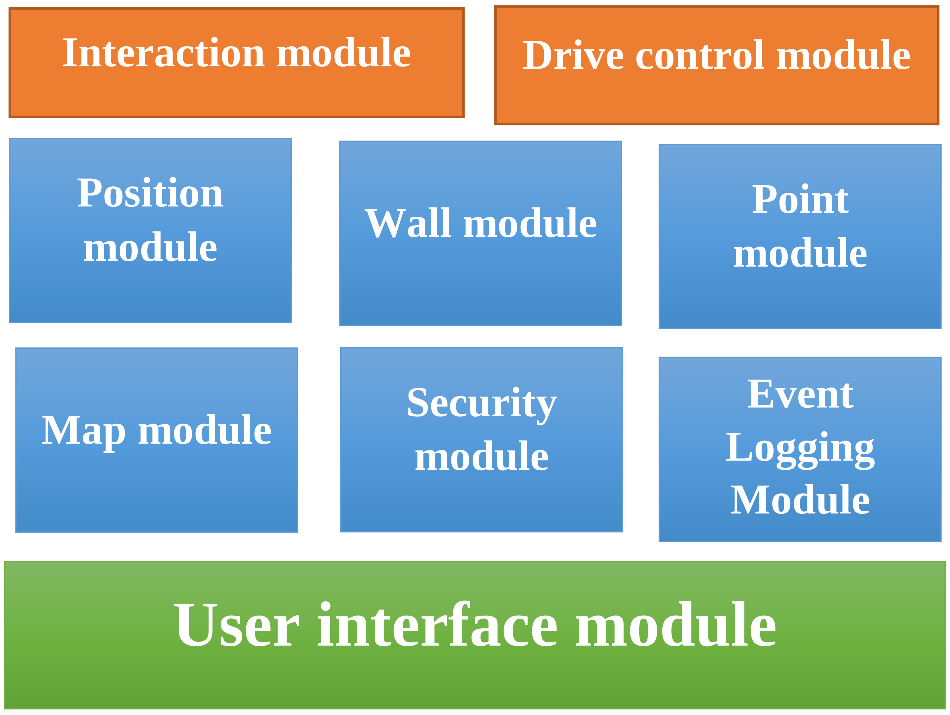


Fig. 17: Control System Architecture.

4.3 SolidWorks

The 3D model of mobile robot was created in SolidWorks. It is a package developed by SolidWorks Corporation (USA) and an application for automated object-oriented design solid models of engineering products [29]. This is the first application of automated design that takes full advantage of the graphical user interface of Microsoft Windows. In SolidWorks is implemented drag-drop functions, so this package is quite easy to learn [29].

Using SolidWorks was developed a model of mobile robot that is presented in Figure 18.

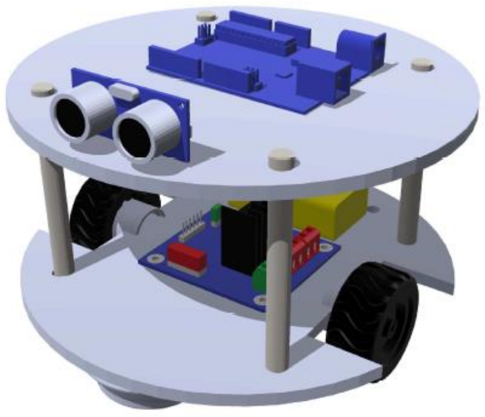


Fig. 18: 3D model of the mobile robot.

The frequency of scans laser rangefinder is 75 Hz. Processing filtering algorithm takes 5.3 milliseconds, which enables to determine the position of the mobile robot and construct a map in real time.

4.4 VREP

Debugging and testing of such a Control System Architecture is challenging. It increases the complexity in the case of debugging on a real robot when bugs are combined with failures of equipment. This task requires a lot of costs not related to the development process: charging the batteries of the robot, the preparation of the working area, download the updated software into the robot, etc. In addition, the robot and its sensors - a rather expensive equipment, so the price of software errors can be very high (e.g., failure of the laser rangefinder when the robot hits an obstacle). Therefore, it is necessary as much as possible to debug and test all the software that is not related to the direct interaction with the hardware to test with the real mobile robot.

There exists a variety of different environments robot emulators such as Microsoft Robotics Developer Studio, Webots, and others. Most of them are commercial and have a closed source code or too complex to be used by people who have no experience with them. It is not possible to do in the simulation of the robot without time or material costs. This opportunity provides a modeling environment Vrep [30]. It is open-source. It is simple to use, description of the objects in it is clear even intuitively, while it provides high flexibility of natural settings and the ability to model arbitrary robots.

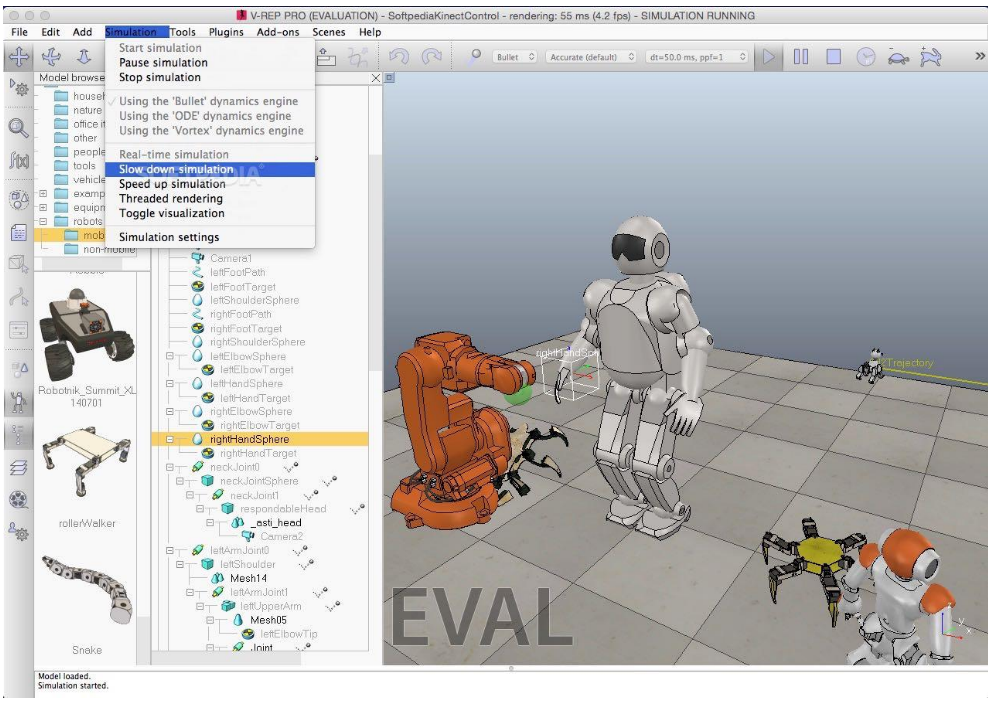


Fig. 19: Vrep simulation environment.

Vrep is a suite of free software for the modeling and control of mobile robots. It allows you to simulate the robot in a given three-dimensional flat world. The package is modeled on the work of many robot sensors including laser range finder and landing gear actuators [30]. User programs can interact with the model through a special library that implements the transfer of control commands and receives information from the simulated sensors via TCP / IP network protocol. The model accurately reflects the behavior of the robot and the laser rangefinder reading when working in an artificial environment.

An architecture mobile robot control system as a distributed system consisting of a single operating system module, processes, allowed to carry out debugging and testing of many modules mainly navigation and control algorithms the robot proposed in Chapters 2 and 3 of the thesis. This required change only the modules that are responsible for interaction with the actuators and sensors of the robot so that they control the movement of models and receive information from virtual sensors of Vrep. Other control modules remain unchanged: the modules that implement the algorithms of navigation and user interface management system operated in this case in the same way as a real robot in the real world.

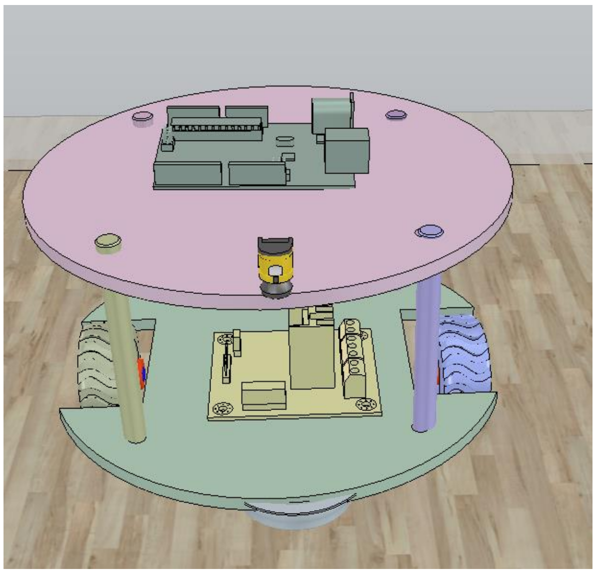
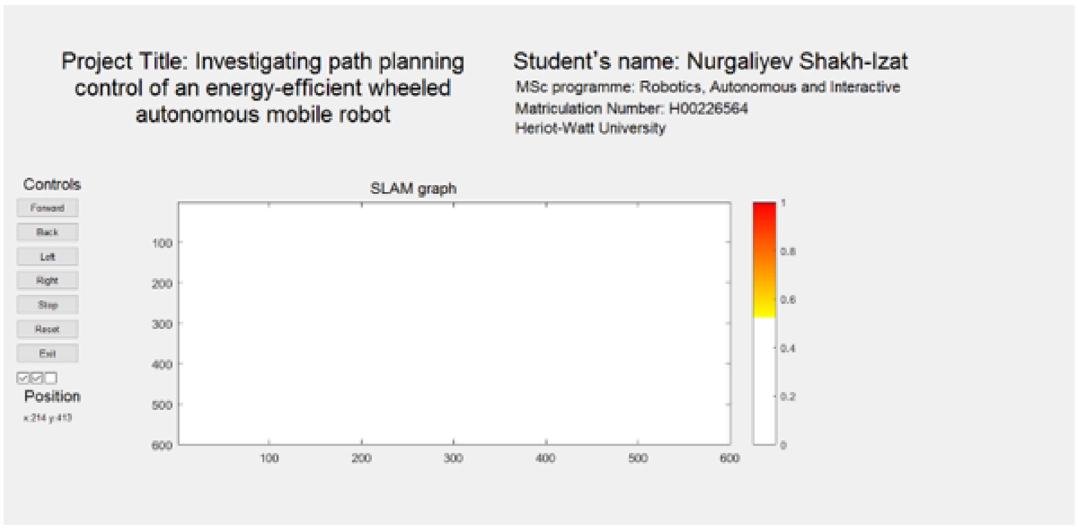


Fig. 20: 3D model of mobile robot that was imported from SolidWorks to Vrep.

Testing the operation of the control system in the Vrep allows you to identify quickly and correct various errors, optimize the developed algorithms and conduct some experiments without the participation of the real robot.

4.5 USER INTERFACE DESIGN

GUI library in MATLAB makes the interface of interaction with the user. In the center of the interface is displayed a graph of the map. Control buttons are located on the left. (Figure 4.4.).


Project Title: Investigating path planning control of an energy-efficient wheeled autonomous mobile robot

Student's name: Nurgaliyev Shakh-Izat
MSc programme: Robotics, Autonomous and Interactive
Matriculation Number: H00226564
Heriot-Watt University

Position
x 214 y 413

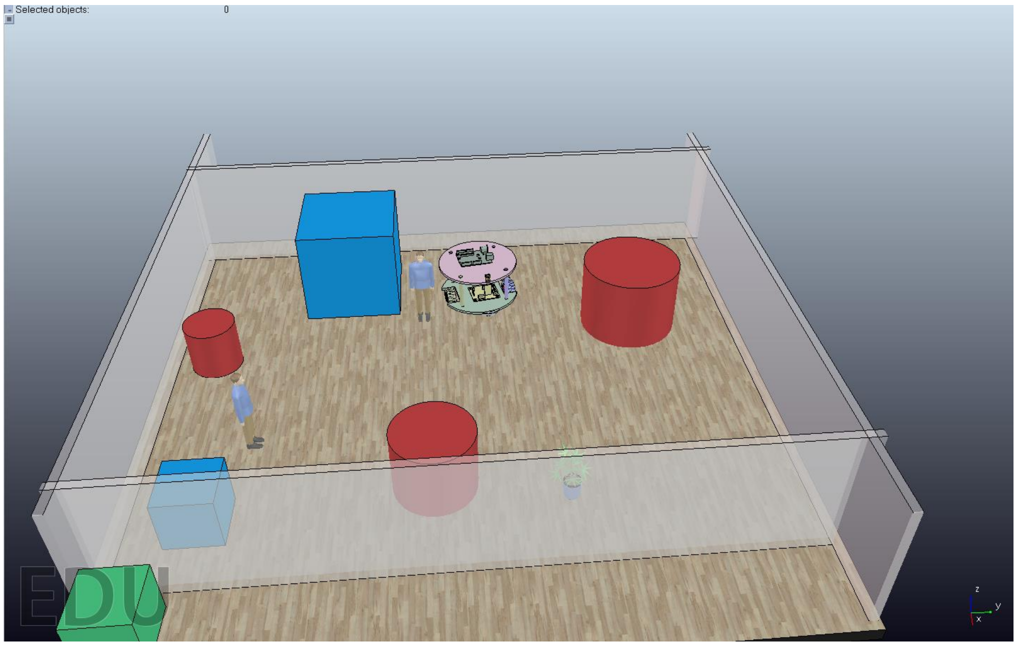
Fig. 21: The user interface of MATLAB application.

The control buttons are used to control the direction of movement of mobile robot (left, right, forward and back). Also, there is stop and exit and reset buttons. Position label determines its position in space. This MATLAB application is used in combination with Vrep.

4.6 EXPERIMENTAL RESULTS

Numerous experiments on the controlling of the robot in the Vrep confirmed the efficiency of navigation and control algorithms. The results are shown below.

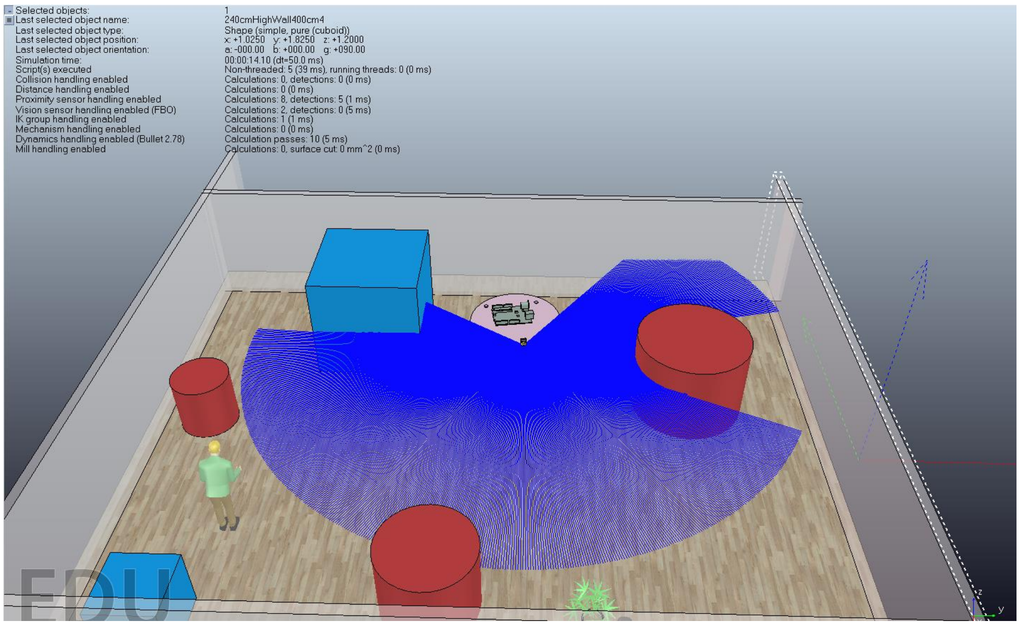
Figure 22 shows a map obtained when controlling a robot model in Vrep. The virtual environment was created randomly. It has dynamic and static obstacles. The robot is controlled manually by using Matlab application. Maximum linear motion robot speed does not exceed 0.8 m / s and the angular 0.6 rad / s.



```
Selected objects:                    0
```

Fig. 22: A virtual environment in Vrep by default.

On-board part is located directly on the mobile robot and provides autonomous operation, and execute commands received. MATLAB application is a part of a portable command center, from which you can remotely control the mobile robot and control the execution of the specified command.



```
Selected objects:                         1
Last selected object name:                240cmHighWall400cm4
Last selected object type:                Shape (simple, pure (cuboid))
Last selected object position:            x: +1.0250   y: +1.8250   z: +1.2000
Last selected object orientation:         a: -000.00   b: +000.00   g: +090.00
Simulation time:                          00:00:14.10 (dt=50.0 ms)
Script(s) executed                        Non-threaded: 5 (39 ms), running threads: 0 (0 ms)
Collision handling enabled                Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                 Calculations: 0 (0 ms)
Proximity sensor handling enabled         Calculations: 8, detections: 5 (1 ms)
Vision sensor handling enabled (FBO)      Calculations: 2, detections: 0 (5 ms)
IK group handling enabled                 Calculations: 1 (1 ms)
Mechanism handling enabled                Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)   Calculation passes: 10 (5 ms)
Mill handling enabled                     Calculations: 0, surface cut: 0 mm^2 (0 ms)
```

Fig. 23: The working principle of laser rangefinder of the mobile robot.

The following figure shows the map obtained by the mobile robot control system in Matlab. SICK S300 rangefinder produces images up to 75 Hz frequency, with 10 mm accuracy, an angular range of 270 ° and line range up to 80m. Laser range finder mounted on the chassis for obtaining scans in the horizontal plane so that the direction of the sensor axis and robot movement direction coincided.

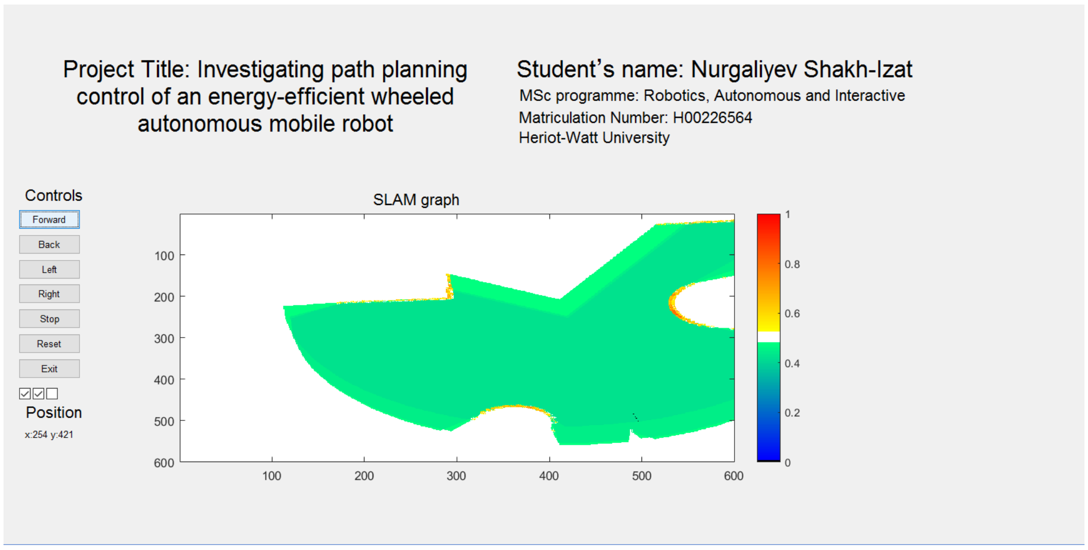

Project Title: Investigating path planning control of an energy-efficient wheeled autonomous mobile robot
Student's name: Nurgaliyev Shakh-Izat
MSc programme: Robotics, Autonomous and Interactive
Matriculation Number: H00226564
Heriot-Watt University
Controls
Forward
Back
Left
Right
Stop
Reset
Exit
Position
x:254 y:421
SLAM graph
100
200
300
400
500
600
1
0.8
0.6
0.4
0.2
0

Fig. 24: SLAM graph of the mobile robot.

For the experiment was introduced three different design with a different set of sensors and navigation algorithms: global navigation method based on wave front algorithm, local navigation with two ultrasonic sensors and local navigation with one laser sensor which is proposed navigation method in this thesis.

In experiments was measured efficiency parameter, expressed as a maximum possible number of repetitions of passing the same map. These parameters were measured in two situations: at the static district map and at dynamically changing the map in which user can add and remove obstacles in the area of the mobile robot. Experimental research has shown the benefits of the proposed method compared to traditional methods of local implementations navigation both on presence of static and dynamic obstacles on the map.

To confirm the performance of the developed algorithm for determining the angular and linear velocity of the mobile robot was used to simulate the process of measurement of the laser range finder located on the mobile robot in MATLAB environment. Experiments on the movement of the robot at a given point in the Vrep simulation environment and real robot confirmed the efficiency of the proposed algorithm in the dissertation.

4.7 SUMMARY OF THE FOURTH CHAPTER

In this chapter of the thesis was presented the results of numerical and field experiments that confirms the theoretical studies in Chapters 2 and 3.

Here was shown the performance of the developed algorithm for determining the angular and linear velocity of the mobile robot with an accuracy sufficient to solve local navigation task - determining the position of the robot by integrating the equations of motion. Maps that was constructed from this localization task has condition to diverge over time. To solve this problem of global navigation is used the particle filter and Bresenham's line algorithm, which allows us to consider some hypotheses about the probable current position of the mobile robot. The possibility of implementing the particle filter by the received position of the robot also demonstrated and proved its efficiency regarding solving the problem of global navigation and constructing of maps, not diverging over time.

Algorithms for navigation and automatic control, proposed in Chapters 2 was applied to create a distributed real mobile robot control system established in the center of the "Robotics" of Heriot-Watt University. The robot is equipped with a laser rangefinder sensor. It can automatically determine the own position and can execute commands of the automatic movement to the point specified by the operator, and return to the previously memorized point. Navigation and robot control using the developed algorithms implemented in real time.

CONCLUSION AND FUTURE RECOMMENDATIONS

In dissertation work the following scientific results are received:

A new method of determining the linear and angular velocities of the robot by solving system of equations describing the movement kinematics of the contour points in the coordinates frame of the laser range finder sensor. This object is achieved by the recursive filtering algorithm, allowing to obtain linear and angular velocities of the robot. It is shown that the results obtained can be applied to solve a local problem of navigating the mobile robot.

A modified method of path planning of mobile robot built by the plan area, providing safe movement of the robot in an environment with obstacles.

The experiments showed the efficiency of the proposed methods of navigation and mobile robot control.

The novelty of this method of mobile robot localization is that it allows you to receive not only the evaluation of the velocity of the robot and its position but also characterized by their error as well. This error is a characteristic problem of localization accuracy of solutions and depends on the nature of the contour and the movement of the mobile robot. The presence of the error is an essential feature of the developed method for determining the position of a mobile robot.

Developed mobile localization and robot control methods have been tested experimentally. Experimental work on the creation of a real mobile robot equipped with a laser rangefinder was conducted in the center of the "Robotics" of Heriot-Watt University. A distributed control system for a mobile robot equipped with a laser rangefinder, which modules implement the proposed algorithms in the dissertation. The established control system has been tested on the computer model of Vrep and the real mobile robot. Developed in the thesis algorithms proved their practical effectiveness of navigation and the control of the mobile robot.

The proposed navigation and control algorithms can be used to create control systems for today's mobile robotic systems equipped with different types of sensors in order to facilitate the work of the operator. The resulting solutions of navigation tasks may run on modern computers in real time and can be used to create the self-contained mobile robotic systems.

REFERENCES

[1] V. Thind, S. Sharma and B. Pandey, Modern Practices in Energy Efficient Electronics. Saarbrücken: LAP LAMBERT Academic Publishing, 2015.

[2] A. Milstein, Improved particle filter based localization and mapping techniques. Waterloo, Ont.: University of Waterloo, 2008.

[3] R. Siegwart and I. Nourbakhsh, Introduction to autonomous mobile robots. Cambridge, Mass.: MIT Press, 2004.

[4] J. Kim, Mobile robot navigation and localization. Saarbrücken: Vdm, 2008.

[5] J. Fernández-Madrigal and J. Blanco Claraco, Simultaneous localization and mapping for mobile robots. Hershey, Pa.: Information Science Reference, 2013.

[6] Y. XU, C. ZHANG and H. XU, "A New Obstacle Avoidance Method for Mobile Robot Based on Laser Range Finder", ROBOT, vol. 32, no. 2, pp. 179-183, 2010.

[7] F. Cuesta and A. Ollero, Intelligent mobile robot navigation. Heidelberg: Springer, 2005.

[8] J. Billingsley and R. Bradbeer, Mechatronics and machine vision in practice. Berlin: Springer, 2008.

[9] G. Sommer and R. Klette, Robot vision. Berlin: Springer, 2008.

[10] T. Bräunl, Embedded robotics. Berlin: Springer, 2008.

[11] D. Sankowski, Computer vision in robotics and industrial applications. Singapore: World Scientific Publishing, 2014.

[12] SLAM using camera and IMU sensors. Washington, D.C.: United States. Dept. of Energy, 2007.

[13] P. Suhm, Vision and SLAM on a highly dynamic mobile two-wheeled robot. [S.l.: s.n.], 2013.

[14] M. Emharraf, M. Bourhaleb, M. Saber and M. Rahmoun, "Mobile Robot: SLAM Implementation for Unknown Indoor Environment Exploration", Journal of Computer Science, vol. 12, no. 2, pp. 106-112, 2016.

[15] T. Kalyan, P. Zadeh, S. Staub-French and T. Froese, "Construction Quality Assessment Using 3D as-built Models Generated with Project Tango", Procedia Engineering, vol. 145, pp. 1416-1423, 2016.

[16] W. Clancey, Working on Mars. Cambridge, Massachusetts: The MIT Press, 2012.

[17] M. Buehler, K. Iagnemma and S. Singh, The 2005 Darpa grand challenge. Berlin: Springer, 2007.

[18] S. Thrun, "Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva", The International Journal of Robotics Research, vol. 19, no. 11, pp. 972-999, 2000.

[19] T. Lefebvre, H. Bruyninckx and J. Schutter, Nonlinear Kalman filtering for force-controlled robot tasks. Berlin: Springer, 2005.

[20] P. Zarchan and H. Musoff, Fundamentals of Kalman filtering. Reston, Va.: American Institute of Aeronautics and Astronautics, 2009.

[21] A. Monjazeb, Autonomous robot navigation based on simultaneous localization and mapping (SLAM), and particle filtering. Ottawa, 2007.

[22] A. Doucet, N. De Freitas and N. Gordon, Sequential Monte Carlo methods in practice. New York: Springer, 2001.

[23]" Human Detecting and Following Mobile Robot Using a Laser Range Sensor", Journal of Robotics and Mechatronics, vol. 26, no. 6, pp. 718-734, 2014.

[24] E. Fotiadis, M. Garzón and A. Barrientos, "Human Detection from a Mobile Robot Using Fusion of Laser and Vision Information", Sensors, vol. 13, no. 9, pp. 11603-11635, 2013.

[25] "Human Following by an Omnidirectional Mobile Robot Using Maps Built from Laser Range-Finder Measurement", Journal of Robotics and Mechatronics, vol. 22, no. 1, pp. 28-35, 2010.

[26] L. Zhou and J. Jiang, "An Approach To Safe Path Planning For Mobile Robot In The Dynamic Environment Based On Compact Maps", *JCP*, vol. 7, no. 2, 2012.

[27] "Using Laser Range Finder and Multitarget Tracking-Learning-Detection Algorithm for Intelligent Mobile Robot", *Sensors and Materials*, 2015.

[28] P. Meel, R. Tiwari and A. Shukla, "Optimization of Focused Wave Front Algorithm in Unknown Dynamic Environment for Multi-Robot Navigation", *International Journal of Intelligent Mechatronics and Robotics*, vol. 3, no. 4, pp. 1-29, 2013.

[29] R. Zhang, W. Wang, Z. Shi, J. Hu, P. Xu, X. Wu and H. Li, "Fire Rescue Robot Design Based on Solidworks", *AMR*, vol. 940, pp. 254-257, 2014.

[30] K. Reddy and K. Praveen, "PATH PLANNING USING VREP", *International Journal of Research in Engineering and Technology*, vol. 02, no. 09, pp. 94-97, 2013.

APPENDIX

```
/**********************************************************
*****************************
 * Author: Nurgaliyev Shakh-Izat   V1.0    August 2016
 * Purpose: Dissertation project
 * Language:  Matlab + Vrep

**********************************************************
****************************/

function slamGui
  Create GUI api.
f =
figure('Visible','off','Position',[360,500,450,285],'Numb
erTitle','off');
% Make the GUI larger
set(gcf, 'units','normalized','outerposition',[0 0 0.8
0.8]);
neoOri = 0;
 % create text labels
mytxt1 = uicontrol('Style','text','String','Student's
name: Nurgaliyev Shakh-Izat',
'Position',[550,490,500,60],'fontsize',20);
mytxt2 = uicontrol('Style','text','String','Project
Title: Investigating path planning control of an energy-
efficient wheeled autonomous mobile robot ',
'Position',[50,450,500,100],'fontsize',20);
mytxt3 = uicontrol('Style','text','String','MSc
programme: Robotics, Autonomous and Interactive Systems
',  'Position',[550,490,500,25],'fontsize',13);
mytxt4 = uicontrol('Style','text','String','Matriculation
Number: H00226564',
'Position',[462,465,500,25],'fontsize',13);
mytxt5 = uicontrol('Style','text','String','Heriot-Watt
University',  'Position',[516,443,300,25],'fontsize',13);
mytxt6 = uicontrol('Style','text','String','SLAM graph',
'Position',[316,373,300,25],'fontsize',13);
mytxt7= uicontrol('Style','text','String','Controls',
'Position',[20,373,80,30],'fontsize',13);
mytxt8= uicontrol('Style','text','String','Position',
'Position',[20,130,80,30],'fontsize',13);

%  Construct the components.
FwdButton =
uicontrol('Style','pushbutton','String','Forward',
'Position',[20,380,70,25],'Callback',{@FwdButton_Callback
});
```

```
BckButton =
uicontrol('Style','pushbutton','String','Back',
'Position',[20,350,70,25],'Callback',{@BckButton_Callback
});
LftButton =
uicontrol('Style','pushbutton','String','Left',
'Position',[20,320,70,25],'Callback',{@LftButton_Callback
});
RghButton =
uicontrol('Style','pushbutton','String','Right','Position
',[20,290,70,25],'Callback',{@RghButton_Callback});
StpButton =
uicontrol('Style','pushbutton','String','Stop',
'Position',[20,260,70,25],'Callback',{@StpButton_Callback
});
RstButton =
uicontrol('Style','pushbutton','String','Reset','Position
',[20,230,70,25],'Callback',{@RstButton_Callback});
ExtButton =
uicontrol('Style','pushbutton','String','Exit',
'Position',[20,
200,70,25],'Callback',{@ExtButton_Callback});
ChkBoxWall =
uicontrol('Style','checkbox','String','Wall',
'Position',[20,
170,15,25],'Callback',{@CheckBox_Callback});
ChkBoxEmpt =
uicontrol('Style','checkbox','String','Empty',
'Position',[35,
170,15,25],'Callback',{@CheckBox_Callback});
ChkBoxDebu = uicontrol('Style','checkbox','String','D',
'Position',[50,
170,15,25],'Callback',{@CheckBox_Callback});
TxtOri = uicontrol('Style','text','String',sprintf(' '),
'Position',[20, 110,70,30],'Max', 4);


global RobotPath1Layer RobotPath2Layer WallLayer
laserScan xW yW neoPose p poseAndTimeUnitTest neoPos
calcPosex calcPosey dT chkLayer

ha = axes('Units','Pixels','Position',[200,100,700,300]);
align([FwdButton,BckButton,LftButton,RghButton,StpButton,
RstButton,TxtOri],'Center','None');


% Create the data to plot.
mapSize = 600; % the size of the map
mapZoom = 60;  % the zoom factor
WallLayer = ones(mapSize,mapSize) / 2; % 0.5 probability
by default
```

```
PrevLayer = ones(mapSize,mapSize);
RobotPath1Layer = ones(mapSize,mapSize) / 2; % 0.5
probability by default
RobotPath2Layer = ones(mapSize,mapSize) / 2; % 0.5
probability by default
neoPose = struct('x', 0, 'y', 0, 'theta', 0); % contains
the position and orientation of neobotix
laserScan = 0;
myColorMap1 = flipud([autumn; ones(6,3); flipud(winter);
zeros(1,3)]); % the color map for blue white red
myColorMap2 = flipud([autumn; ones(6,3); ones(65,3); ]);
% the color map for white red
probailityConstant = 0.95; % the constant of probability
chkLayer = [true(1) true(1) false(1) false(1)]; % Wall
Empty Path1 Path2 layers

%% Initialize the GUI.
% Change units to normalized so components resize
automatically.
set([f,ha,FwdButton,BckButton,LftButton,RghButton,StpButt
on,RstButton,TxtOri,ExtButton,ChkBoxWall,ChkBoxEmpt,ChkBo
xDebu],'Units','normalized');
%Create a plot in the axes.
image(WallLayer);
% Assign the GUI a name to appear in the window title.
set(f,'Name','slam gui','NumberTitle','off')
% Move the GUI to the center of the screen.
movegui(f,'center')
% Make the GUI visible.
set(f,'Visible','on');

% Connect to V-REP (if not already connected)
if(exist('vrep','var') == 0)
    [vrep, clientID] = connectVREP('127.0.0.1',19997);
end
[~,motorLeft] = vrep.simxGetObjectHandle(clientID,
'wheel_left#0', vrep.simx_opmode_oneshot_wait);
[~,motorRight] = vrep.simxGetObjectHandle(clientID,
'wheel_right#0', vrep.simx_opmode_oneshot_wait);
[~,sickHandle] = vrep.simxGetObjectHandle(clientID,
'SICK_S300_fast#0', vrep.simx_opmode_oneshot_wait);
[~,neoHandle0] = vrep.simxGetObjectHandle(clientID,
'neobotix#0', vrep.simx_opmode_oneshot_wait);
% Initailize the timer
tic
% Initialize the main figure
DrawAllLayer()
SetWheelSpeed(0,0);
currentTime = toc;
GetPose();
poseAndTimeUnitTest = double([]);
```

```
calcPosex = neoPos(1);
calcPosey = neoPos(2);
% Set the checkboxes to checked
set(ChkBoxWall ,'Value',1);
set(ChkBoxEmpt ,'Value',1);

%% Push button and checkbox callbacks

function FwdButton_Callback(~,~)
  SetWheelSpeedToTarget(2,2, 4);
end

function BckButton_Callback(~,~)
  SetWheelSpeedToTarget(-2,-2, 4);
end

function LftButton_Callback(~,~)
  SetWheelSpeedToTarget(-0.5,0.5,25);
end

function RghButton_Callback(~,~)
  SetWheelSpeedToTarget(0.5,-0.5,25);
end

function StpButton_Callback(~,~)
  SetWheelSpeed(0,0);
end

function RstButton_Callback(~,~)
  WallLayer = ones(mapSize,mapSize) / 2; % 0.5
probability by default
  RobotPath1Layer = ones(mapSize,mapSize) / 2; % 0.5
probability by default
  DrawAllLayer()
end

function UndButton_Callback(~,~)
  WallLayer = PrevLayer;
  colormap(myColorMap1)
  image(WallLayer,'CDataMapping','scaled');
  colorbar
  caxis([0,1])
end

function ExtButton_Callback(~,~)
  disconnectVREP(vrep, clientID);
  close(f);
end

function CheckBox_Callback(~,~)
  DrawAllLayer()
```

```
end


%% Nested functions
function SetWheelSpeed(left, right)
  DrawAllLayer()
  vrep.simxSetJointTargetVelocity(clientID, motorLeft,
left, vrep.simx_opmode_oneshot_wait);
  vrep.simxSetJointTargetVelocity(clientID, motorRight,
right, vrep.simx_opmode_oneshot_wait);
end

function SetWheelSpeedToTarget(left, right, targetAngle)
  SetWheelSpeed(left, right) % start the robot
  i = 0;
  posMotor = 0;
  turns = 0;
  maxIteration = 200;
  %targetAngle = 8;
  prevPos = [0 0]; % contains the actual (2) and the
previous position of the wheel (1)
  while (abs(turns*pi*2+posMotor) < abs(targetAngle) && i
< maxIteration) % the robot moves until it reaches the
target
       [~, posMotor] =
vrep.simxGetJointPosition(clientID, motorRight,
vrep.simx_opmode_oneshot_wait);
       i = i + 1;
       prevPos(1) = prevPos(2);
       prevPos(2) = posMotor;
       if mod(i,10) == 0
           GetPose();
           prevTime = currentTime;
           currentTime = toc;
           dT = currentTime - prevTime;
           calcPosex = calcPosex + left * 5
*(cos(neoPose.theta) * dT*1000);
           calcPosey = calcPosey + left * 5
*(sin(neoPose.theta) * dT*1000);
           [tx, ty] = ConvertXYToLayer(calcPosex,
calcPosey);
           WallLayer(tx+0, ty) = 0;
           poseAndTimeUnitTest = [poseAndTimeUnitTest;
double(neoPose.x), double(neoPose.y),neoPose.theta,
double(neoPos(1)), double(neoPos(2)), currentTime];
           set(TxtOri, 'String', sprintf('x:%d %d\ty:%d
', neoPose.x, tx, neoPose.y));
       end
       if prevPos(1) > 0 && prevPos(2) < 0 % if the wheel
reaches the
           turns = turns + 1;
```

```
            DrawAllLayer()
        end
    end
    SetWheelSpeed(0, 0) % stop the robot
end

function [xMap, yMap] = ConvertXYToLayer(xReal, yReal)
    xMap = int64((xReal+11)*mapZoom); % convert neoPos to
matrix element
    yMap = int64((yReal+1)*mapZoom); % convert neoPos to
matrix element
end

function GetPose()
    [~,neoPos] = vrep.simxGetObjectPosition(clientID,
sickHandle, origoHandle, vrep.simx_opmode_oneshot_wait);
    [~,neoOri] = vrep.simxGetObjectOrientation(clientID,
neoHandle0, origoHandle, vrep.simx_opmode_oneshot_wait);
    % convert neoOrientaion
    if neoOri(1) < 0
      neoOri(2) = neoOri(2) * - 1 + pi/2;
    else
      neoOri(2) = neoOri(2) + pi + pi/2;
    end
    [neoPose.x, neoPose.y] = ConvertXYToLayer(neoPos(1),
neoPos(2));
    neoPose.theta = neoOri(2);
    currentTime = toc;
end

function GetLaserScannerData()
    res = 19;
    while (res~=vrep.simx_return_ok)

[res,laserScan]=vrep.simxReadStringStream(clientID,'measu
redDataAtThisTime0', vrep.simx_opmode_streaming);
    end
    laserScan = vrep.simxUnpackFloats(laserScan);
    laserScan = reshape(laserScan,3,size(laserScan,2)/3);
    if size(laserScan,2) > 684 % todo
      laserScan = laserScan(:,end-684:end);
      laserScan = [laserScan(1,:) ; (laserScan(2,:) .* -1);
(laserScan(3,:))]; % flip laser scanner data
      %plot(laserScan(1,:), laserScan(2,:), 'ro')
    end
    FilterLaserScanner();
end

function AddRobotToLayer()
    GetPose();
    for n = 1:10
```

```
      xD = neoPose.x + int64(n*cos(neoPose.theta));
      yD = neoPose.y + int64(n*sin(neoPose.theta));
      if (xD < mapSize && yD < mapSize && xD > 0 && yD >
0) % if they fit into the matrix
        WallLayer(xD, yD) = 0.9;
      end
  end
  %RobotPath1Layer(neoPose.x,neoPose.y) = 100;
end

function AddWallToLayer()
  if size(laserScan,2) > 684 % todo
   for i = 1:size(laserScan, 2)
    xW = neoPose.x + int64(mapZoom*laserScan(1, i));
    yW = neoPose.y + int64(mapZoom*laserScan(2, i));
    if (xW < mapSize && yW < mapSize && xW > 0 && yW > 0)
% if they fit into the map
      WallLayer(xW, yW) = 1-((1 - WallLayer(xW, yW)) *
probailityConstant); % increase the probability
      WallLayer(xW+1, yW+1) = 1-((1 - WallLayer(xW, yW))
* probailityConstant); % increase the probability
      WallLayer(xW-1, yW-1) = 1-((1 - WallLayer(xW, yW))
* probailityConstant); % increase the probability
      WallLayer(xW-1, yW+1) = 1-((1 - WallLayer(xW, yW))
* probailityConstant); % increase the probability
      WallLayer(xW+1, yW-1) = 1-((1 - WallLayer(xW, yW))
* probailityConstant); % increase the probability
    end
   end
  end
end

function AddEmptyToLayer()
    GetPose();
    if get(ChkBoxDebu,'Value') == 0
        a = emptyLaser(laserScan, neoPose.theta, mapZoom,
probailityConstant, 0);
    else
        a = emptyLaser(laserScan, neoPose.theta, mapZoom,
probailityConstant, 1);
    end
    [rowsBig, columnsBig] = size(WallLayer); % mapSize
    [rowsSmall, columnsSmall] = size(a)
    % Specify upper left row, column of where
    % we'd like to paste the small matrix.
    row1 = neoPose.x- ((mapZoom+1)*4*1.6) % todo
    column1 = neoPose.y - ((mapZoom+1)*4*1.6)
    % Determine lower right location.
    row2 = row1 + rowsSmall - 1
    column2 = column1 + columnsSmall - 1
    if row1 <= 1
```

```
            a = a(-row1+2:end, :);
            row1 = 1
        end
        if column1 <= 1
            a = a(:, -column1+2:end);
            column1 = 1
        end
        % If row is out from the map
        if row2 >= mapSize
            row2 = mapSize
            a = a(1:mapSize - row1 + 1, :);
        end
        % If column is out from the map
        if column2 >= mapSize
            column2 = mapSize
            a = a(:, 1:mapSize - column1 + 1);
        end
        % See if it will fit.
        if row2 <= rowsBig
            % It will fit, so paste it.
            WallLayer(row1:row2, column1:column2) =
    WallLayer(row1:row2, column1:column2) .* a;
        end
    end

    function points = CalcLine(x1, y1, x2, y2) % Calculates
    the laser beam lines (empty area)
        % bresenham algorithm [was take from internet]
        % Bresenham algorithm is incremental scan conversion
    algorithm
        x1=round(x1); x2=round(x2);
        y1=round(y1); y2=round(y2);
        dx=abs(x2-x1);
        dy=abs(y2-y1);
        steep=abs(dy)>abs(dx);
        if steep t=dx;dx=dy;dy=t; end
        %The main algorithm goes here.
        if dy==0
            q=zeros(dx+1,1);
        else
            q=[0;diff(mod([floor(dx/2):-dy:-
    dy*dx+floor(dx/2)]',dx))>=0];
        end

        %and ends here.
        if steep
            if y1<=y2 y=[y1:y2]'; else y=[y1:-1:y2]'; end
            if x1<=x2 x=x1+cumsum(q);else x=x1-cumsum(q); end
        else
            if x1<=x2 x=[x1:x2]'; else x=[x1:-1:x2]'; end
            if y1<=y2 y=y1+cumsum(q);else y=y1-cumsum(q); end
        end
        points = [x y];
        points = points(2:end, 1:end); % exclude the wall
    itself from the empty area
    end

    function DrawAllLayer()
        % Wall Empty Path1 Path2 layers
        chkLayer(1) = get(ChkBoxWall,'Value');
        chkLayer(2) = get(ChkBoxEmpt,'Value');
      set(TxtOri, 'String', sprintf('x:%d y:%d', neoPose.x,
    neoPose.y));
        GetPose();
        GetLaserScannerData();
        laserScan = [cos(neoPose.theta),-
    sin(neoPose.theta),0;sin(neoPose.theta),cos(neoPose.theta
    ),0;0,0,1] * laserScan; % rotate laser scanner data
    (orientation)
        PrevLayer = WallLayer;
        if chkLayer(1)
            AddWallToLayer()
        end
        if chkLayer(2)
            AddEmptyToLayer()
            colormap(myColorMap1)
        else
            colormap(myColorMap2)
        end
        if chkLayer(3)
            AddRobotToLayer()
        end
        image(WallLayer,'CDataMapping','scaled');
        h = zoom; % zoom by default
        caxis([0,1])
        set(h,'Motion','both','Enable','on'); % zoom by
    default
        colorbar

    end

    function FilterLaserScanner()
        % filtering the own contour of the robot from the
    laser scanner measurement (if not inside the 0.3 radius
    circle)
        filteredLaser = NaN(size(laserScan));
        i = 0;
        for n = 1:size(laserScan, 2)
            if (sqrt(laserScan(1,n)^2 + laserScan(2,n)^2) >
    0.3)
                i = i + 1;
                filteredLaser(:,i) = laserScan(:,n);
            end
        end
        laserScan = filteredLaser; % todo
    filteredLaser(:,1:i); jobb lenne, csak akkor 685-re kell
    figyelni
    end
    end
```