Intel Serv Robotics
DOI 10.1007/s11370-016-0194-5

ORIGINAL RESEARCH PAPER

An autonomous stereovision-based navigation system (ASNS) for mobile robots

Khalid Al-Muteb[1] · Mohammed Faisal[1] · Muhammad Emaduddin[1] · Mohammed Arafah[1] · Mansour Alsulaiman[1] · Mohamed Mekhtiche[1] · Ramdane Hedjar[1] · Hassan Mathkoor[1] · Mohammed Algabri[1] · M. A. Bencherif[1]

Received: 31 August 2015 / Accepted: 13 February 2016

Abstract Recently, stereovision has appeared in robotics as a source of information for real-time mapping and path planning. In this paper, an intelligent motion system for mobile robots is designed and implemented using stereovision. The proposed system uses stereovision as a primary method for sensing the environment, and the system is able to navigate intelligently in an indoor environment with varying degrees of obstacle complexity. It creates noiseless and high-confidence 3D point clouds and uses these point clouds as an input for the mapping and path-planning modules. The proposed system was built by developing, enhancing, and integrating various techniques, modules and algorithms. The Stereovision-based Path-planning module is the integration of three main enhanced techniques: (1) the multi-baseline multi-view stereovision filter (MMSVF), (2) accurate floor detection and segmentation (AFDS), and (3) the intelligent gazing module (IGM). This Stereovision-based Path planning (MMSVF, IGM, and AFDS) was integrated with the Fuzzy Logic Motion Controller (FLMC). All techniques, modules and algorithms are implemented using a multi-threaded and client–server-based architecture. To prove the viability and robustness of our proposed system, we have integrated all components of the system into a fully functional mobile robot navigation system. We compared the performance of the main modules with that of similar modules in the literatures, and showed that our modules had better performance. Testing the whole system is more important than just testing each module individually. To the best of our knowledge, the literatures lack such testing. Hence, in this paper we present the performance of our complete integrated system in different environments using different parameters and different architectures.

Keywords Mobile robot · Stereovision · 3D point clouds · Multi-baseline · Fuzzy logic

1 Introduction

Vision-guided mobile robotics (VGMR) systems are evolving with each passing year due to rapid gains in computational power and the reliability of vision sensors that can be deployed on mobile platforms. The primary focus of VGMR research is to allow a mobile robot to navigate in an unstructured environment without collision. In recent years, several researchers have looked at methods for setting up autonomous mobile robots for navigation tasks. Among these methods, stereovision-based navigation is a promising approach for reliable and efficient navigation. Stereovision cameras collect 3D data at a high frame rate and can also capture color and texture as opposed to the relatively poor performance of laser scanners. In this paper, the stereovision camera is used for mapping and path planning. In addition, the laser device and ultrasonic sensors are integrated to avoid nearby dynamic obstacles during the navigation. The stereovision part of this paper focuses on solving problems that arise from shortcomings in local stereo-matching algorithms such as the management of specular reflections and the lack of discriminative image features and repetitive patterns [1] within an indoor environment. Many local stereo-matching methods strive to remain within the realm of real-time algorithms while remaining accurate, e.g., AdaptGCP [2], AD-Census [3] and SAD-IGMCT [4], to name a few. Instead of presenting a completely new local stereo-matching tech-

✉ Mohammed Faisal
Mfaisal@ksu.edu.sa

[1] Robotics Lab, College of Computer and Information Sciences (CCIS), King Saud University, Riyadh, P.O. Box 5117, 11543, Saudi Arabia

Published online: 17 March 2016

nique, we propose a higher level filtering method that aims at minimizing the noise and patchy 3D information generated by local stereo-matching techniques such as the Sum of Absolute Difference, i.e., SAD. This higher level filtering approach makes the proposed method suitable for filtering point clouds generated by any local stereo-matching techniques. To prove the viability and robustness of our proposed filter, we have integrated our filter into a fully functional mobile robot navigation system. A multi-baseline stereovision camera by Point Grey is used as the primary input sensor for our proposed method. In this paper, an intelligent motion system for mobile robots is designed and implemented. The system is able to achieve an autonomous navigation within an unstructured dynamic indoor environment. The system has two main characteristics: it uses stereovision for mapping and path planning and intelligent control for navigation. In this system, we have designed and developed many systems and modules and used them within the proposed system. The most important parts of the system are the multi-baseline multi-view stereovision filter, the stereovision-based gaze control strategy, RANSAC segmentation-based floor detection and the fuzzy logic motion controller (FLMC) for indoor navigation. The resultant system has the following features: (a) stereovision-based navigation within an unknown dynamic indoor environment, (b) adaptability to indoor lighting conditions via stereo-matching parameters and tuning of point cloud filtering parameters, (c) the capability to utilize previously built maps for navigation, (d) tuning of path-planning and localization parameters, (e) a laser measurement sensor-assisted system for emergency braking and minor path adjustment, and the capability of moving through multiple waypoints while performing fully featured navigation. We showed that the system could successfully navigate autonomously in different environments (tight spaces and small rooms, open spaces, and corridor). We compared the performance of the system in these environments. The performance parameters were accuracy, execution time, and average robot speed.

This paper is organized as follows. In Sect. 2, a literature review is presented. The autonomous stereovision-based navigation system (ASNS) is explained in Sect. 3; the components of ASNS are presented in Sect. 4; Sect. 5 explains the system experimentation and performance; robot speed performance is presented in Sect. 6; real-time execution and analysis of ASNS is presented in Sect. 7; comparison with other works is presented in Sect. 8, and the conclusion is given in Sect. 9.

2 Related work

Our stereo-based navigation system involves and integrates many techniques, such as environment sensing, obstacles and floor detection, mapping, and robot localization and navigation. Therefore, we divided the related work into the following categories: feature point detection, obstacle detection, accurate floor detection, multi-baseline and stereovision, use of active stereovision for map enhancement, and navigation operation.

2.1 Feature point detection

A good feature point detector is absolutely essential for vision-based navigation [5]. The 3D point cloud retrieved via stereovision can be further filtered to detect good feature points. An accurate approach used for the mapping application is to detect 3D feature points within the point cloud provided by the stereovision camera. Well-known techniques for such purposes are point feature histograms (PFH) [6], NARF points [7], spin-image [8] and eigen-CSS [9]. These feature points help in the process of "point cloud registration" [6] and thus result in point clouds that are well aligned (stitched), although these have been gathered at temporally and spatially varying poses of a stereovision camera. If point cloud registration is performed accurately, the resultant point cloud can be used for effective navigation; otherwise, a common problem referred to as the "sliding-problem" [6] renders the overall mapping process erroneous. 2D features do not translate into 3D features but are similar in terms of their persistence through the angle of approach, illumination conditions, scale invariance and robustness to noise. 2D features such as KLT, SIFT, SURF and Harris-Corners [9–12] are widely used in 2D image processing. Corresponding 3D points for 2D points are widely used in the literature as good feature points, but success and results are limited in terms of successful mapping applications [5,13].

2.2 Obstacle detection

An additional issue in stereovision-based navigation is obstacle detection. The authors in [14] gave an introduction to an obstacle detection approach based on stereovision that is proposed for mobile robot navigation. The aim of the approach is to detect a real environment to allow a mobile robot to find a safe path even in complex scenarios. The novelty is represented in terms of the two-stage perception structure. The detection stage infers the relationship between obstacles and the ground and determines the region of interest. Based on this, the confirmation stage focuses on characterizing the contours and positions of obstacles and removing the majority of artifacts. A vision-equipped apelike robot based on the remote-brained approach was presented by Masayuki et al. [15]. They present a new type of robot that has two arms and two legs like an ape and is meant to study a variety of vision-based behaviors. The robot does not bring its own brain within the body. It leaves the brain in the mother

environment and talks with it by radio links. The brain is raised in the mother environment, which is inherited over generations. In this framework, the robot system can have a powerful vision system in the brain environment. They have applied this approach toward the formation of a vision-based dynamic and intelligent behavior of a multi-limbed mobile robot.

2.3 Accurate floor detection

Many methods emphasize the accuracy of 3D reconstruction [1], but ground plane detection using minimal or noisy 3D data [2] and detection of dominant planes in the environment is a detailed area in itself [3]. Among the surveys, very few works suggest a method that detects both dominant and subdominant planes along with the height-based classification of obstacles. Among other obstacle detection methods, some [4,16] completely ignore the significance of detecting the ground plane and variations within it, while others [17] use pixel-based region segmentation and classification techniques that may or may not be able to classify floor anomalies.

2.4 Multi-baseline and stereovision

Multi-baseline stereo has been used in the past to optimize grid maps. The first practical use of occupancy grids can be attributed to Elfes [18]. Another use of occupancy grids was applied by Murray et al. [19]. However, it suffers from typical stereovision-based mapping shortcomings including sensitivity to false positives when environment observation time is short (due to a slower convergence rate while updating occupancy probability). In [1], an interesting occupancy grid mapping technique is presented that uses both dense and sparse point clouds from Stereovision and SLAM (simultaneous localization and mapping), respectively (structure and motion). In this case, although the algorithm appears to address false positives relatively well, there is a weakness in encountering outliers, as no multi-view, multi-baseline stereo-matching technique is utilized. Several other techniques were found in the literature that attempt to utilize multi-baseline stereo at the level of local stereo-matching or multi-view stereo to filter noise, e.g., [20–23], but no technique was found to use a higher level multi-baseline filtering or to use it in combination with multi-view stereo filtering.

2.5 Use of active stereovision for map enhancement

A highly enriched body of literature exists regarding gaze techniques using various forms of movable camera mounts. Bio-inspired camera gaze generation approaches top the list, with noticeable works listed in [24–26]. Many of these works focus on the ultimate goal of achieving human-head-motion-like performance. Camera mounts bearing a variety of degrees of freedom (DOF) configurations are employed in these works. Some authors use vergence [24,27], zoom, focus, aperture or baseline variance techniques, but these techniques fail to render the flexibility and tight coupling of gaze generation with the SLAM and path-planning modules. Some authors, such as Clady [26] and Nakagawa [28], developed a vision system that has a pan tilt zoom (PTZ) camera to track a moving object to enable the robot to perform smooth pursuit of a target. In [29], however, a more relevant localization and 3D information gain-sensitive gaze generation strategy is discussed. The authors present a gaze strategy that strikes a balance between keeping the maximum possible number of features within the field of view (FOV) and adjusting the camera pose so that maximum obstacle exploration can occur. The approaches deployed in [30,31] use a gaze generation decision system that works on the basis of a utility function that balances the cost of exploring new terrain with the utility of a changing camera pose toward future positions that maximize information gain.

2.6 Navigation operation

Despite the advances in the field of autonomous robotics, many problems still exist. Many of the difficulties are due to the unknown dynamic environment. Various useful techniques, such as fuzzy logic, genetic algorithms, and neural networks are used to address an unknown and dynamic environment. A fuzzy logic for indoor navigation has been presented in [32]. The authors proposed how to use fuzzy logic control for target tracking control of the wheeled mobile robot (WMR). The authors focused on the navigation without recognizing the obstacles; they simply used FLC for motion control of the WMR. An online navigation for WMR is presented in [33]. In this paper, the authors used two fuzzy logic controls to navigate the scout2 robot in an unknown dynamic environment. The tracking fuzzy logic controller is used to navigate the WMR to its target, and the obstacles avoiding fuzzy logic controller is used to avoid the obstacles. An indoor navigation system using fuzzy logic control is presented in [34]. The authors used the camera and fuzzy logic to move the robot toward its goal. However, the authors concentrated on the navigation without taking into account obstacle avoidance; they just use FLC for navigation. A real-time fuzzy logic control scheme for target tracking by autonomous mobile robots is proposed in [34]. This scheme used infrared sensors and dual robots; the first WMR is the moving target, and the second is the tracker [35]. In addition to fuzzy logic control, a genetic algorithm and neural network have been used to improve the control scheme. Fuzzy logic control and the genetic algorithm are also used in [36] to find the optimal parameters for the fuzzy logic. Four fuzzy logic controllers are used to navigate the mobile robot in [37]. These four fuzzy controls form a hierarchical control. Three fuzzy controllers

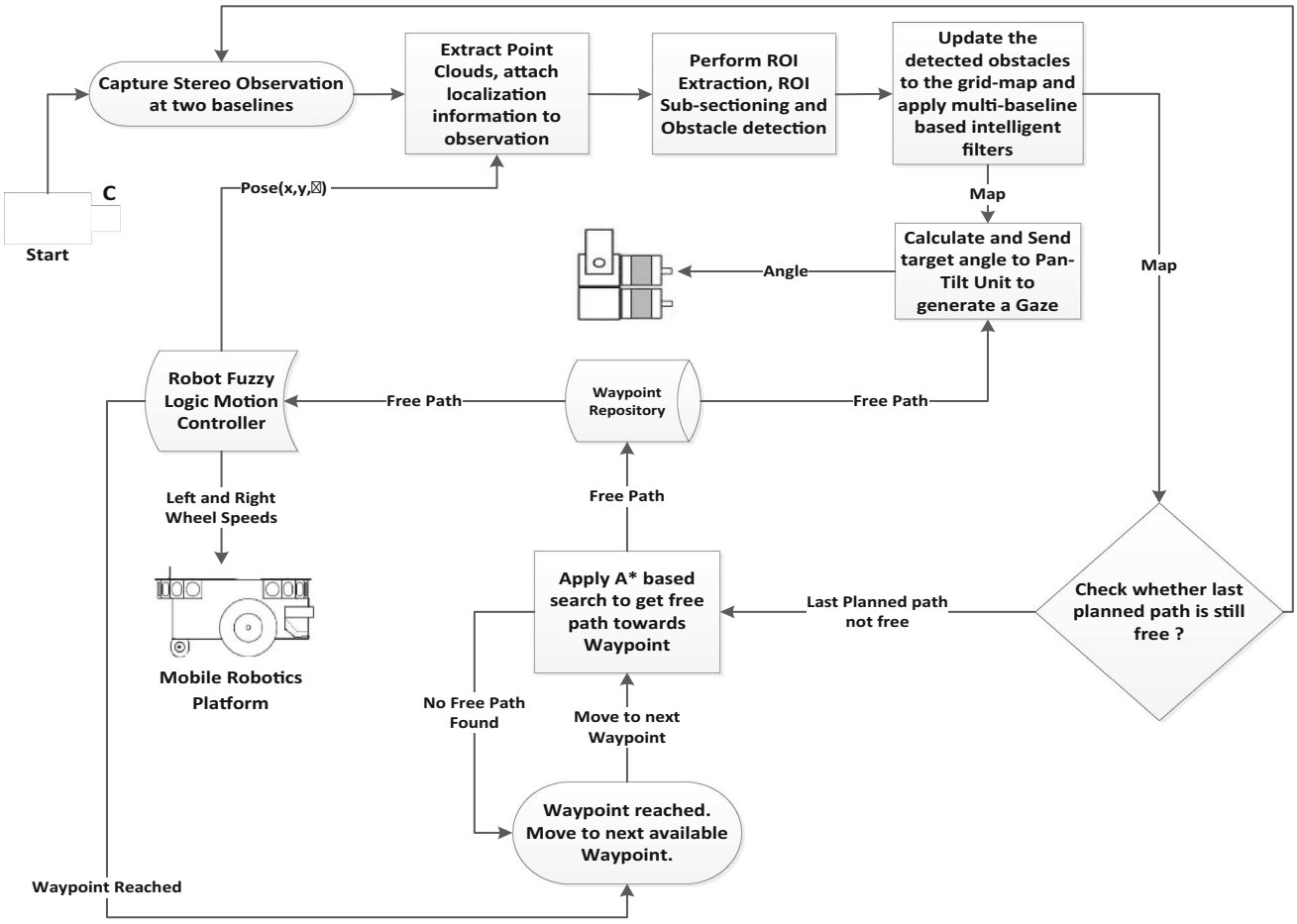


Fig. 1 Flowchart for ASNS

have been used for navigation and obstacle avoidance. One of them is used as a supervisory controller.

3 The proposed autonomous stereovision-based navigation system (ASNS)

In this section, we are going to give an overview of the ASNS system and the interaction between its components, and in the next section, we will explain each component in detail. As we have mentioned, ASNS was built by integrating various algorithms and techniques built by the authors. The stereovision-based path-planning module (MMSVF, IGM, and SAFDS) was integrated with FLMC. Multiple third-party pre-implemented algorithms (A* path planning, Sum of Absolute Differences stereo-matching algorithm, statistical outlier removal algorithm, band-pass filtering) were integrated to form a complete system. A flowchart for the resultant system is displayed in Fig. 1. ASNS needs many parameters and data to initiate the navigation process. The parameters are the path-planning parameters, point cloud filtering parameters, stereo-matching algorithm parameters, stereovision observation filtering parameters and the list of waypoints (multiple target points). The system starts by capturing multi-baseline stereovision observations gathered using a Bumblebee stereovision camera. The 3D points representing the environment are extracted from these observations. Basic stereo-matching filters are applied at this stage. These filters include the texture validation mapping filter, uniqueness validation filter, back and forth validation filter [38], and surface validation filter [39]. These point clouds are down-sampled to achieve real-time processing speed. Finally, a region of interest (ROI) that ignores points above a height of 1.2 m is selected and is passed onto the ROI sub-sectioning sub-module. This sub-sectioning module is necessary to reduce the time complexity required to perform point cloud manipulation. We were able to reduce time complexity for various filtering algorithms. Next, the sub-sectioned point clouds are passed over to the filtering sub-module. This sub-module filters the 3D point cloud for false positives that arise due to specular reflections and direct exposure to highly bright indoor lights by our self-developed MMSVF. These filtered point clouds are then passed through an AFDS-based plane-fitting method to detect the floor and

obstacles. Obstacles as small as wires, low-profile obstacles and significant carpet deformities were successfully detected by our self-developed floor segmentation method. The detected obstacles are then accurately updated on a global grid map. This grid map is used to plan the paths using the A* algorithm when a path is requested by any other module within the system. In this case, a set of x and y coordinates called "Waypoints" are handed over to a Fuzzy Logic Motion Control Module FLCM via a "Waypoint Repository". A "Waypoint Repository" is a file formatted in a pre-agreed format that serves as a communication channel between FLMC and the rest of the system. These points include the starting points, the target and all the intermediate waypoints that represent the free path between the robot's current location and the target location. The motion module (MM) of the FLCM is used to navigate the robot to its target using the waypoints. However, in the case of nearby dynamic obstacles (if the distance between the robot and the obstacles is <50 cm), the obstacles avoidance module (OAM) is used to avoid the obstacles. If the dynamic obstacles are distant, the stereovision-based path-planning module will generate a new path and send it to FLCM. The free path and the map are also forwarded to a "Gaze Generation Module". This module is responsible for generating exploratory gazes toward the free path and obstacles near the robot, based on the intelligent gazing module. It must be noted that the observation cycle is running continuously, and the map is updated after each observation. This is performed to ensure the accuracy of the map in case a path-replanning request is received from any module.

4 Components of the ASNS system

The ASNS system consists of many modules. Each module executes specific tasks. The interaction between all these modules represents the whole system. ASNS is implemented using a multi-threaded and client–server-based architecture, as illustrated in Fig. 2. In this section, we are going to discuss in detail each component and module in the ASNS system.

ASNS is divided into the following nine components or modules:

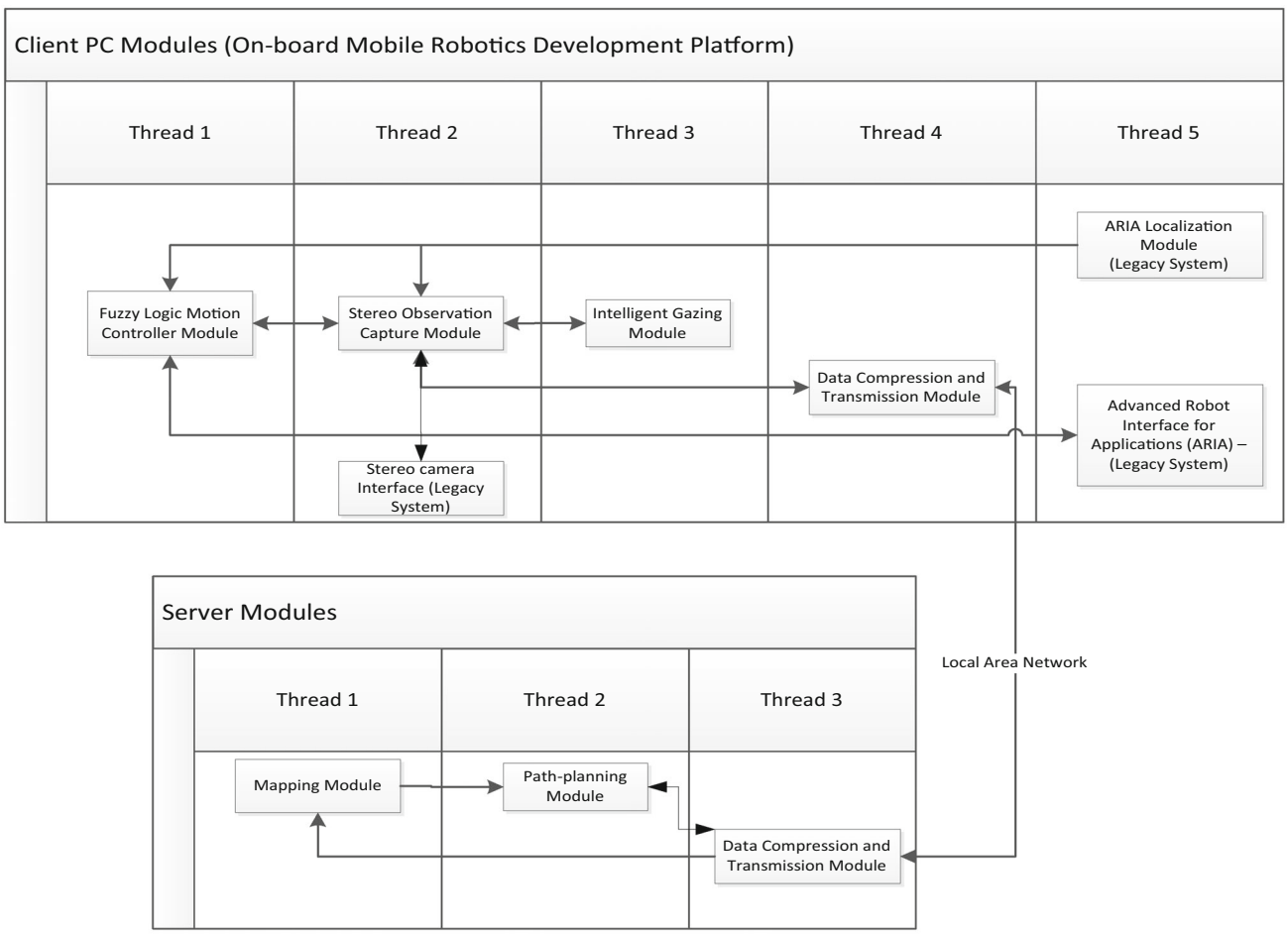


Fig. 2 Architecture diagram for ASNS

(A) Stereovision Initialization Module (SVIM)
(B) Multi-baseline Multi-view Stereovision Filter (MMSVF) Module
(C) Accurate Floor Detection and Segmentation (AFDS)
(D) Stereo-Observation Capture Module (SOCM)
(E) Observation Read and Write Module (ORWM)
(F) Mapping Module (MM)
(G) Path-Planning Module (PPM)
(H) Fuzzy Logic Motion Controller (FLMC) Module
(I) Intelligent Gazing Module (IGM)

The client application onboard robot consists of a stereo camera interface, a mobile robot control and communication interface, a point cloud compression and communication module, an ARIA-based robot localization legacy module, a stereo-observation capture module and an intelligent gazing module. Parallel processing is employed via distributing all the modules over multiple separate OS level threads (displayed in Fig. 2). A GPU is also deployed over the client side that assists in algorithm speed-up via the ArrayFire library interface. Third-party libraries used in our system, i.e., the mobile robotics programming toolkit [40] (MRPT) and point cloud library (PCL), that serve as an execution backbone of several algorithms also employ GPU cores independently.

The server application consists of mapping, path planning and point cloud compression and communication. Parallel processing is deployed both via Operating System and GPU threads over the server machine. Distribution of modules over the Operating System level threads can be observed in Fig. 2. Third-party libraries deployed at the server end also include the MRPT and PCL.

4.1 Stereovision initialization module (SVIM)

This module is the startup module for the system. This module initiates the system execution by starting up the stereo-observation capturing loop. Inputs to this module include a host of parameters as listed in Table 1.

4.2 Multi-baseline multi-view stereovision filter (MMSVF)

The module is implemented using client–server architecture. A thin application works on the client side (mobile robot), while another heavier server-side application works on the server. The thin application is used to capture multi-view 3D point clouds with two distinct baseline configurations (wide (24 cm) and narrow (12 cm)) and to extract the ROI. Then, compress the ROI point clouds and send them to the server. After the capturing process, two ROIs are extracted from these point clouds (P' : ROI for a narrow baseline z-axis: 0.1–2.5 m, and P'' : ROI for a wide baseline z-axis: 2.5–5.0 m. Using these baselines, the FOV of the robot will be up to 5 m as illustrated in Fig. 3.

The Bumblebee camera pose $Pose_{\mathrm{BB}}$ and the robot pose $Pose_{\mathrm{Rob}}$ are associated with each point cloud observation. This enables the method to reliably deduce localization information. The point clouds received from the client application are fed into an *AFDS* module. This module is employed to distinguish the floor from the obstacles. Thus, for two point clouds (P', P''), after the execution of the plane-fitting technique, two stochastic 2D occupancy grid maps (G', G''), will be populated, respectively. To perform multi-baseline consolidation, each vertex $U_{x,y}$ belonging to (G', G'') is loaded with further information in addition to the usual probability value, i.e.,

$$U_{x,y} = \begin{cases} \mathrm{prob}_{u_{xy}} \\ \mathrm{centroid\_}xyz_{u_{xy}} \\ \mathrm{updatestatus}_{u_{xy}} \\ \mathrm{confidence}_{u_{xy}} \end{cases} \quad \ldots \tag{1}$$

where

$$\mathrm{prob}_{u_{xy}\,\mathrm{at\,time}\,t} = \log\left(\frac{p\left(U_{x,y} = \mathrm{occupied}|p_1, \ldots, p_t\right)}{1 - p\left(U_{x,y} = \mathrm{occupied}|p_1, \ldots, p_t\right)}\right) \ldots \tag{2}$$

Equation 2 implements a binary Bayes Filter. Here, $Prob_{u\,x\,y\,\mathrm{attimet}}$ is represented in log odds form [41] for time t. $p\left(U_{x,y} = \mathrm{occupied}|p_1, \ldots, p_t\right)$ represents the occupancy probability given the measurement p (p qualifies as a point that belongs to an obstacle). In Eq. 1, updatestatus indicates that the particular vertex of the grid map received an update from the current observation (currently received point cloud from the sensor). centroid$_xyz_{u_{xy}}$ contains the centroid location of all 3D points contributing to the cell's occupancy probability. The confidence value can have three possible states, i.e., low, medium and high.

The consolidation process is as follows:

1. At the beginning of the consolidation process, set $\mathrm{confidence}_{u_{xy}}$ for all vertices to low.
2. For all the vertices $U_{x,y}$ in (G', G''), which receive updates in the form of single or multiple points belonging to an obstacle, the value of $\mathrm{confidence}_{u_{xy}}$ is set to medium.
3. For all the vertices $U_{x,y}$ in G' *or* G'', for which the distance from the robot center is less than 1.5 m and which also receive updates in the form of single or multiple points belonging to an obstacle, the value of $\mathrm{confidence}_{xy}$

Table 1 SVIM input parameter

SVIM input parameter	Description of the parameter
Set of waypoints	A set of waypoints (target points) that constitutes the intended visit-plan for the robot
Image size/resolution	The size of the image that is to be used as the initial gray-scale image
Disparity range	The range with which distance measurements are to be performed
Mask size	The coarseness of features that are to be matched between images
Preprocessing	Whether matching should be performed on gray-scale or preprocessed images, such as edge images
Surface validation parameters	The parameters for the Surface Validation algorithm to verify the correctness of matched-between images
Regions of interest	The region of the image on which processing should be performed. This feature allows processing speed-up
Sub-pixel interpolation	Establishes correspondences to sub-pixel accuracy allowing generation of more precise distance measurements
Statistical outlier removal algorithm parameters	The threshold parameters to perform sparse outlier analysis and removal on stereovision-based point clouds
Size of grid map	The maximum size of the grid map used for path planning should be defined by the user
Resolution of the grid map	The dimension of each cell of the grid map must also be defined by the user. The usual dimensions used in the experiments were 5×5 cm or 10×10 cm or 15×15 cm
Robot radius	To add more safety as the robot steers through an indoor environment, the user is given an option to specify the virtual radius of the robot for path-planning purposes
Occupancy threshold	The user specifies a value between 0 and 1 that helps the path-planning module to decide the values to consider as an obstacle within the grid map. For example, a value of 0.4 will determine that all values less than 0.4 must be considered as an obstacle within the grid map

is set to high. This is performed because a wide baseline-based stereo output produces very inaccurate 3D points for the initial 1.5 m in front of the camera for the current configuration of stereo-parameters that have been initialized for the system.

4. Now, for two vertices *a and b*, where

   $a \in G', b \in G''$,

   do the following

   $$if\ (\text{prob}_a = \text{medium})\ AND\ (\text{prob}_b = \text{medium})$$
   $$\{\quad \text{prob}_a = \text{high};$$
   $$\text{prob}_b = \text{high};\ \}$$

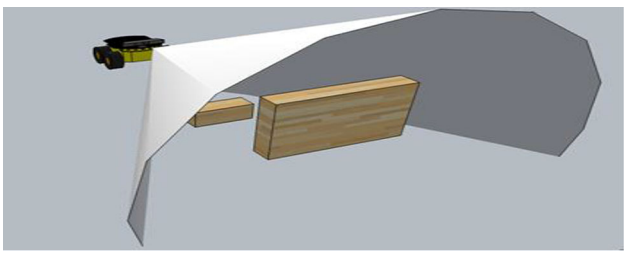


Fig. 3 The visibility cone for the proposed module

5. Add vertices $U_{x,y}$ to the final consolidated grid map G for which $\text{prob}_{u_{xy}}$ is high.

Fig. 4 Bumblebee XB3 multi-baseline stereovision camera by Point Grey Research, Inc

Table 3 MMSVF output

MMSVF output	Description of the parameter
High-confidence grid map cells	MMSVF determines the grid-cells that are most likely to have obstacles. Such grid-cells are called high-confidence grid map cells

These ROI point clouds are decompressed and reconstructed at the server to project and merge the multi-grid maps. In this module, we used the Bumblebee XB3 camera, displayed in Fig. 4, for the multi-baseline stereovision capture. The input parameters for MMSVF are listed in Tab. 2. The output for MMSVF is listed in

Table 3. Details regarding the MMSVF are given in our self-developed filter [42].

4.3 Accurate floor detection and segmentation (AFDS)

AFDS is a recursive RANSAC segmentation-based algorithm that estimates the dominant and subdominant plane models for all the navigable planes within a detected floor or a ground plane. The algorithm also divides the input point clouds intelligently into multiple regions of interest for both efficiency and accuracy enhancement. The recursive estimation approach for determining plane parameters helps to detect multiple planes within each region. Among other benefits of this approach, reduction of search space size for the estimation of plane parameters stands out as the most striking result of this work. This region-wise plane estimation approach also helps to reduce the computational load by selectively dropping less significant floor sections from the estimation process. The floor estimation technique coupled with sensor response functions for two different point cloud generators further investigates the robustness of the method when deployed on two distinct sensors, i.e., the RGB+D sensor and a stereovision camera. In our experiments, we segment navigable floor planes in real time using a slowly moving sensor. The location and geometrical parameters of the floor planes are stored in a database that serves to recreate the 3D environment and also helps in optimizing the plane-fitting problem. The planes are associated to a grid map, which serves as a path-planning reference to the mobile robot in our experiments. The results of the floor detection and the precision of floor anomaly detection are compared sensor-wise and with the ground truth defined by obstacle heights and configuration. The real-time performance of the proposed algorithm was analyzed using a well-documented RGB-D SLAM dataset. The steps of AFDS are displayed in Fig. 5, and the details regarding the AFDS are given in [43].

4.4 Stereo-observation capture module (SOCM)

This module consists of many steps. The first step initializes the stereo-parameters, the second step is the acquisition of raw stereo images, and the third step is image rectification and stereo-matching via the sum of absolute intensity differences (SAD) algorithm. The fourth step is to apply a band-pass filter, which is applied to the resultant 3D point cloud to filter out the unnecessary data. The creation of camera and stereo-processing context is performed during every iteration for each of the two baselines. This particular step is the most time-consuming step in the whole system, as we will see in Sect. 7.

Table 2 MMSVF input parameter

MMSVF input parameter	Description of the parameter
Obstacle detection range for narrow baseline	The filter needs this range as the Bumblebee camera narrow baseline can detect obstacles between 0.9 m and approximately 10 m. A narrow baseline is well suited to obstacles less than 5 m away
Obstacle detection range for wide baseline	The filter needs this range as the wide baseline of the bumblebee camera can detect obstacles between 0.9 m and approximately 10 m. The wide baseline is well suited to obstacles farther than 3 m
Overlap range between both baselines	User needs to decide the overlap range in which many of the noisy points reside. Usually, the majority of noisy points lies between 3 and 6 m

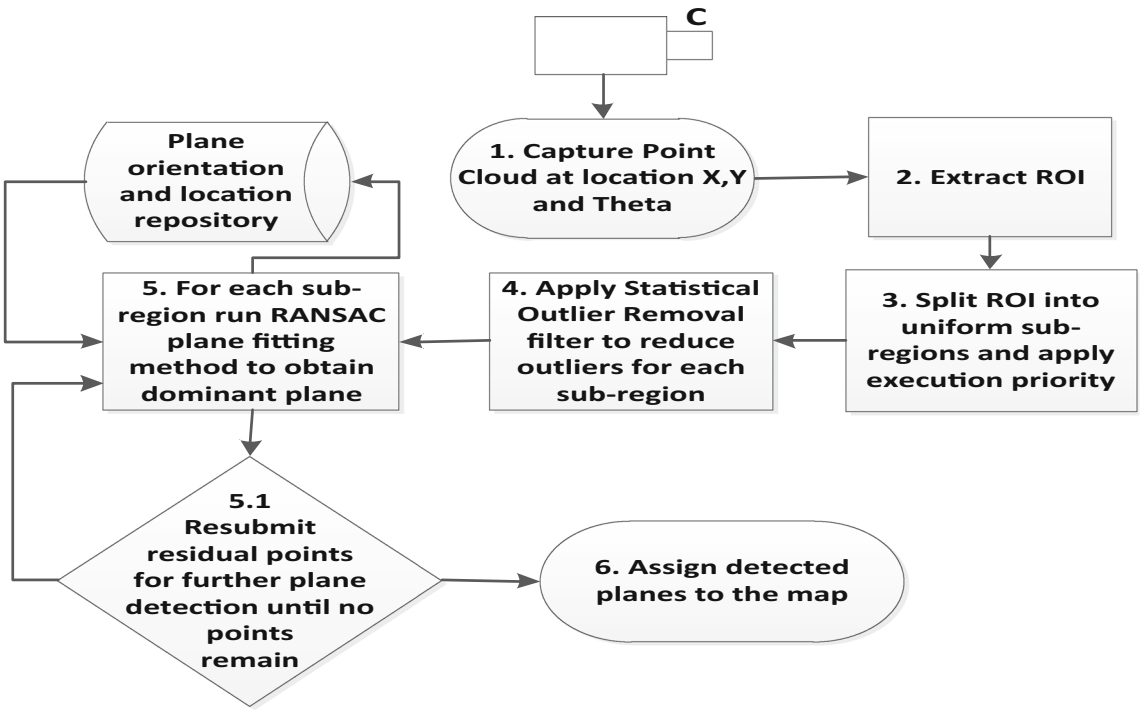


Fig. 5 Floor detection and segmentation flowchart

4.5 Observation read and write module (ORWM)

This module comprises two sub-modules that execute in an inter-leaved fashion. One sub-module named the "Socket-based communication sub-module" consists of socket initialization for the purpose of communication between modules that reside on the server and those that reside on the robot. This sub-module also sends and receives socket-based signals between server and client (robot) for coordinating shared file access. The other sub-module is known as the "file-based observation write and read module". This module consists of file-read and file-write statements. These statements transfer stereo-observation data between the server and the client (robot).

4.6 Mapping module (MM)

The mapping module is responsible for mapping 3D observations from stereovision cameras to a 2D map. This mapping must be reliable; thus, a series of band-pass, statistical outlier removal and other custom-designed filters must be applied to minimize false positives. An accurate pose is required from the localization module for observation integration into the map. The resultant 2D map is path-planning ready and is updated after the capture of each observation by the SOCM. A binary Bayes filter-based map update algorithm, detailed in [42], is responsible for the map update process.

4.7 Path-planning module (PPM)

The path-planning module uses the Streaming SIMD Extensions 2 (SSE2) optimized A* algorithm for path-planning purposes. This algorithm requires the current 2D location of the robot, the radius of the robot and the 2D target location to be able to compute the free path. Path planning is performed when obstacle information in the map is updated in a way that affects the planned path. Path planning is performed for the first time between the current position of the robot and the target point while assuming that the entire map consists of free space. Later, with the collection of camera observations, obstacles are inserted in the map, which may or may not require path planning. The output of PPM is listed in Table 4.

4.8 Fuzzy logic motion controller (FLMC)

In this module, three fuzzy logic controllers are integrated to navigate the mobile robot in an environment cluttered with obstacles. A configuration setting module (CSM) is used to rotate the mobile robot when an error is detected within the current heading and the target heading. The motion module (MM) is used to navigate the robot to its target. The obstacles avoidance module (OAM) is used for obstacle avoidance behavior for nearby and dynamic obstacles. This module adjusts the individual speeds for the robot wheels in a real-time control loop.

Table 4 PPM output

Output of PPM	Description of the parameter
Set of planned intermediate waypoints until the next waypoint	PPM generates the set of planned intermediate waypoints that represent the free path between the current robot position and the upcoming waypoint. The difference between waypoints and planned intermediate waypoints is that the former is provided by the user, while the latter represents the free path generated by the SVIM
Updated grid map	SVIM generates a raw bitmap representing the values within the grid map. This bitmap resolution depends upon the resolution of the grid map used within SVIM. In a particular case, each grid map cell represents 25 cm^2. A 30×30 m area will be represented by a bitmap image of 600×600 pixels

The inputs of MM are the angle between the direction of the robot to the target and the x-axis (error angle) and the distance between the robot and the target. The outputs of MM are the velocities of the left and right motors. MM has been implemented using seven membership functions for both inputs (the angle error and distance error). Left velocity LV and right velocity RV of the motors are the output of the MM. LV and RV in the MM have been implemented using seven membership functions.

OAM is used for avoiding static and dynamic obstacles (the distance between the obstacle and robot is less than 20 cm). The inputs of OAM are the distance between the left, front, and right sides of the robot and the obstacles (LD, RD, and FD, respectively). These distances are acquired using laser devices and ultrasonic sensors. We use the laser to take advantage of its high accuracy and the ultrasonic sensor to take advantage of the higher coverage area for any obstacle. LD is from degree 50 to 30, FD from degree -30 to 30, and RD from degree -50 to -30. The outputs of the OAM module are the velocities of the left LV and the right RV of the motors. LV and RV in AM have been implemented using three membership functions. CSM is used to overcome the problem of the existence of close intermediate points (if the distance between the robot and the point is <50 cm). CSM is used to rotate the robot before the motion. The input of CSM is the rotate angle at which the robot should rotate. The outputs of CSM are the velocities of the left and right motors. A flowchart for the FLMC is displayed in Fig. 6. Details regarding the FLMC are given in our self-developed controller [44].

The input and the outputs parameters of FLMC are listed in Tables 5 and 6, respectively.

4.9 Intelligent gazing module (IGM)

This module is used to determine the optimal rotation angle for the stereovision camera during navigation. This module consists of two sub-modules. One module consists of statements that send commands to a pan tilt unit (PTU) via the serial port and introduces time delays to let the motion of the PTU complete toward a desired gaze angle. These gaze angles are computed in the other module that performs this computation on the basis of planned-path points and obstacle locations within the map. The active vision module is deployed to achieve detection and tracking of moving objects, in addition to mapping applications. We have used active vision as a way to maximize FOV as well as to observe critical obstacles to optimize path planning [45]. The path-planning module of our navigation system is tightly coupled with the gaze control system (GCS). This is performed to make the stereovision camera observe the upcoming path ahead of the mobile robot. In addition, the GCS is able to direct the camera toward the most critical obstacles that have a high impact on future path-planning decisions.

The intelligent gazing module has the following features that have a significant impact on the overall system.

- The major goal of this module is to enhance obstacle avoidance while moving toward the target point via the shortest path under a given scenario.
- This module improves mapping accuracy and thus consequently improves path-planning accuracy as well.
- The gaze is focused toward the immediately planned path in front of the robot during path-specific gaze behavior. This ensures that the free path is constantly checked for the presence of any dynamic obstacles.
- The gaze is also focused on the closest obstacle detected within the range of a 0.9–5 m radius. This ensures that most critical obstacles are observed for the maximum possible amount of time.

Details regarding the IGM are given in the self-developed modules [45].

5 System experimentation and performance

The system was tested using PowerBot as the robot platform. PowerBot is equipped with four different types of sensors: laser range finder, sonar sensor, bumper sensor, and stereovision, as shown in Fig 7. The system path-planning and execution performance was tested successfully many times for different navigation tasks that were as long as

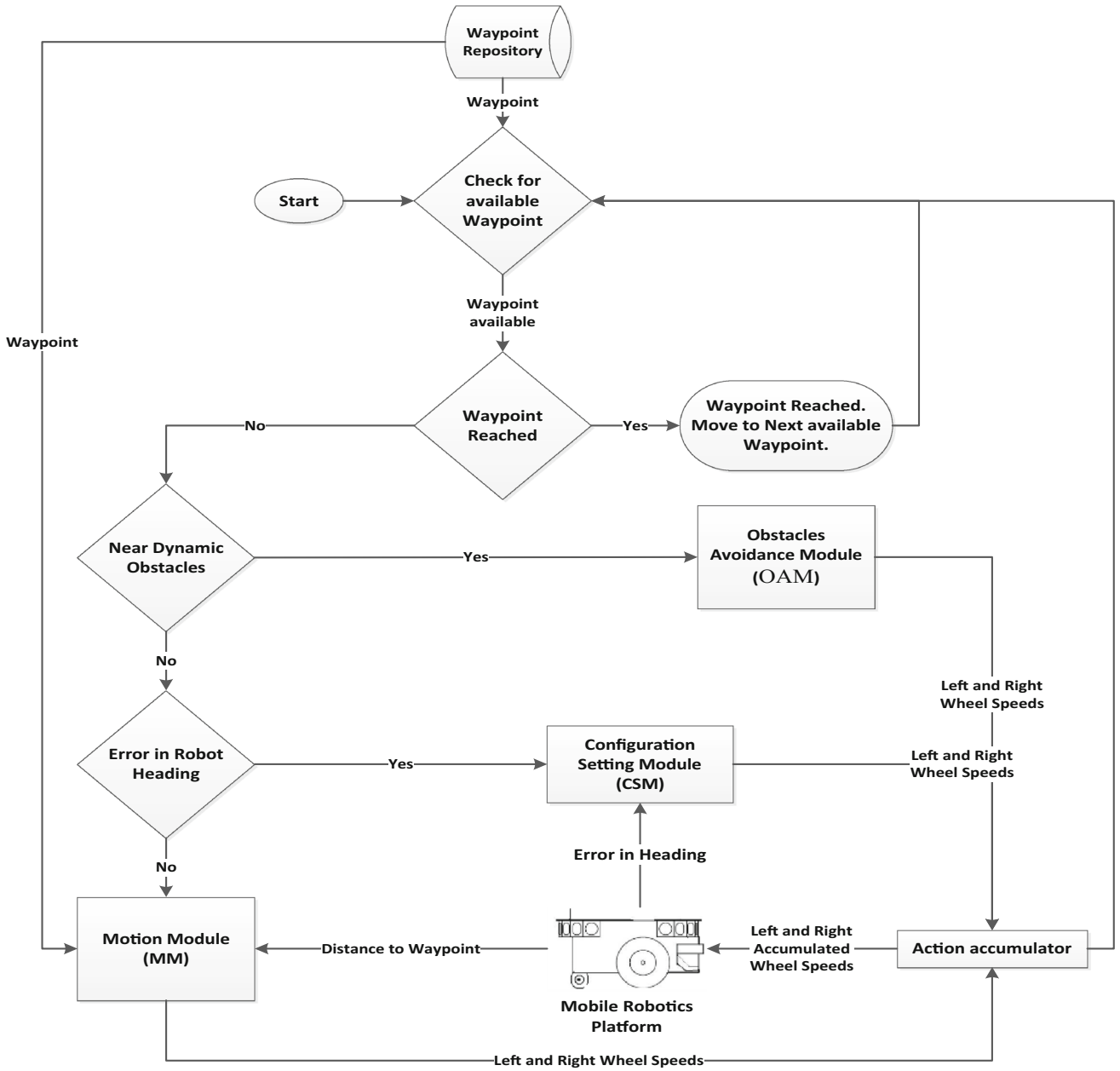


Fig. 6 Detailed flowchart of fuzzy logic motion controller

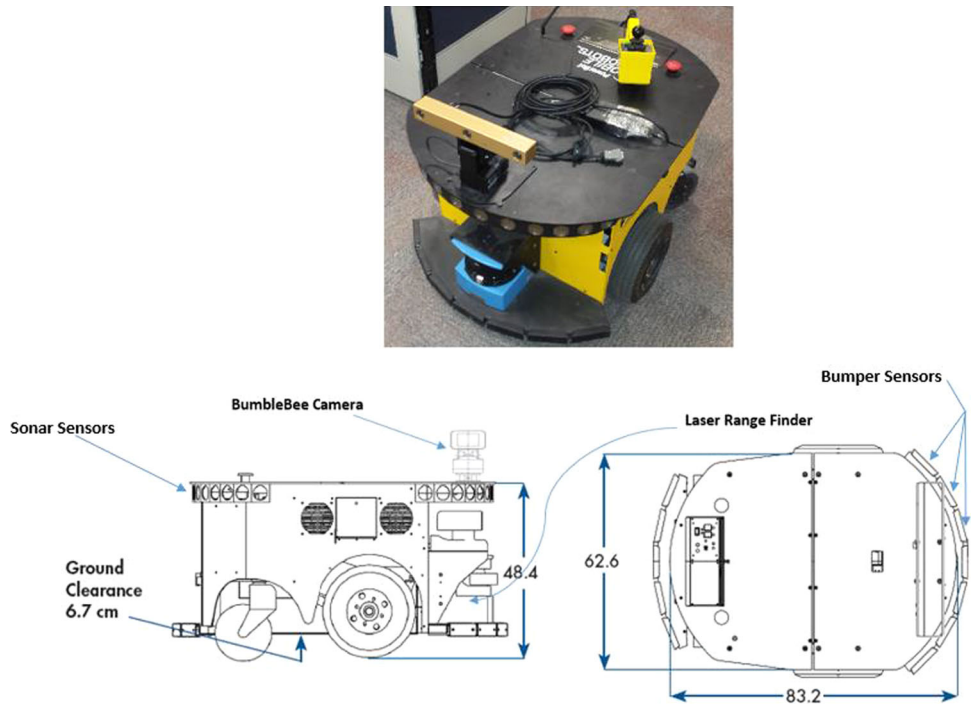
Table 5 FLMC input parameter

FLMC input parameter	Description of the parameter
Set of waypoints	FLMC requires a set of planned intermediate waypoints (target points) from the SVIM
Threshold distance for obstacle detection for laser measurement sensor	The user specifies range values in centimeters for the front laser sensor cone, right laser cone and left sensor cone Only objects detected in this range are considered obstacles

Table 6 FLMC output

FLMC output	Description of the parameter
Left wheel and right wheel speed commands	Speeds of the left and right wheels of the robot. These speeds directly contribute to a smooth trajectory-following and obstacle avoidance behavior
Localization	The robot kinematic model and a higher level soft computing are used to determine the robot pose after the motion commands are executed on the robot

Fig. 7 Robot platform



25 m (the available environment). The environment for the experimentation was unknown, dynamic and indoors with varying degrees of obstacle complexity. A snapshot of the results captured from some of these experiments is displayed in Fig. 8. Figure 8 illustrates the path-planning and trajectory-following outputs of ASNS. As observed, the robot moved from the start point to the target point in corridor "a", inside room "b", and in open indoor space "c" using the proposed system (ASNS). During the movement, many techniques, modules and algorithms (SVIM, MMSVF, AFDS, SOCM, ORWM, PPM, FLMC, and IGM) interacted as illustrated previously. The Red arrows indicate the camera gaze angle, the blue circle indicates the actual robot path, and the distance is in centimeters. The navigation performance indicators of 20 trials for the "Average System Iteration time" are listed in Table 7. During the movement of the robot, when the path is free of obstacles or when the robot is near to the target, it starts to gaze backward to explore the whole environment to be ready for future navigation.

The experimental result indicates that large obstacles tend to be detected much earlier than smaller ones, thus reducing the need for path re-planning. The "Average System Iteration Time" parameter represents the time consumed in between and includes the observation capture from the Stereo-Observation Capture Module and the generation of motion on a free path from FLMC. This parameter is highly dependent upon the number of 3D points generated by the SOCM. The time required to filter and map the observation is directly proportional to the number of incoming 3D points for each observation. Open space environments with few obstacles required the least average iteration time, while small rooms with a relatively high number of obstacles required the maximum average iteration time.

Table 8 shows the accuracy with respect to ground truth of the proposed system. As can be seen in Table 8, the maximum error of all scenarios was 0.05 m.

6 Robot speed performance

The "Average Robot Speed" in Table 9 was generally recorded to be the highest for open spaces with a low number of obstacles. The same parameter was noted as the lowest for an environment cluttered with a medium number of obstacles within tight spaces and small rooms.

The parameter has fewer values for experiments involving a medium number of obstacles for both open spaces and tight spaces because smaller obstacles are detected after a significantly larger number of stereo-observations compared to the large number of obstacles. Such a situation gives rise to more frequent path re-planning. There is an aberration to the preceding statement, i.e., when experiments are conducted within the corridor environment, slightly large average speed values were recorded. This happens because the system design limits the camera focus to the obstacles ahead instead of generating frequent exploratory gazes. Such behavior results in increased observation time for a medium number of obstacles lying on the corri-

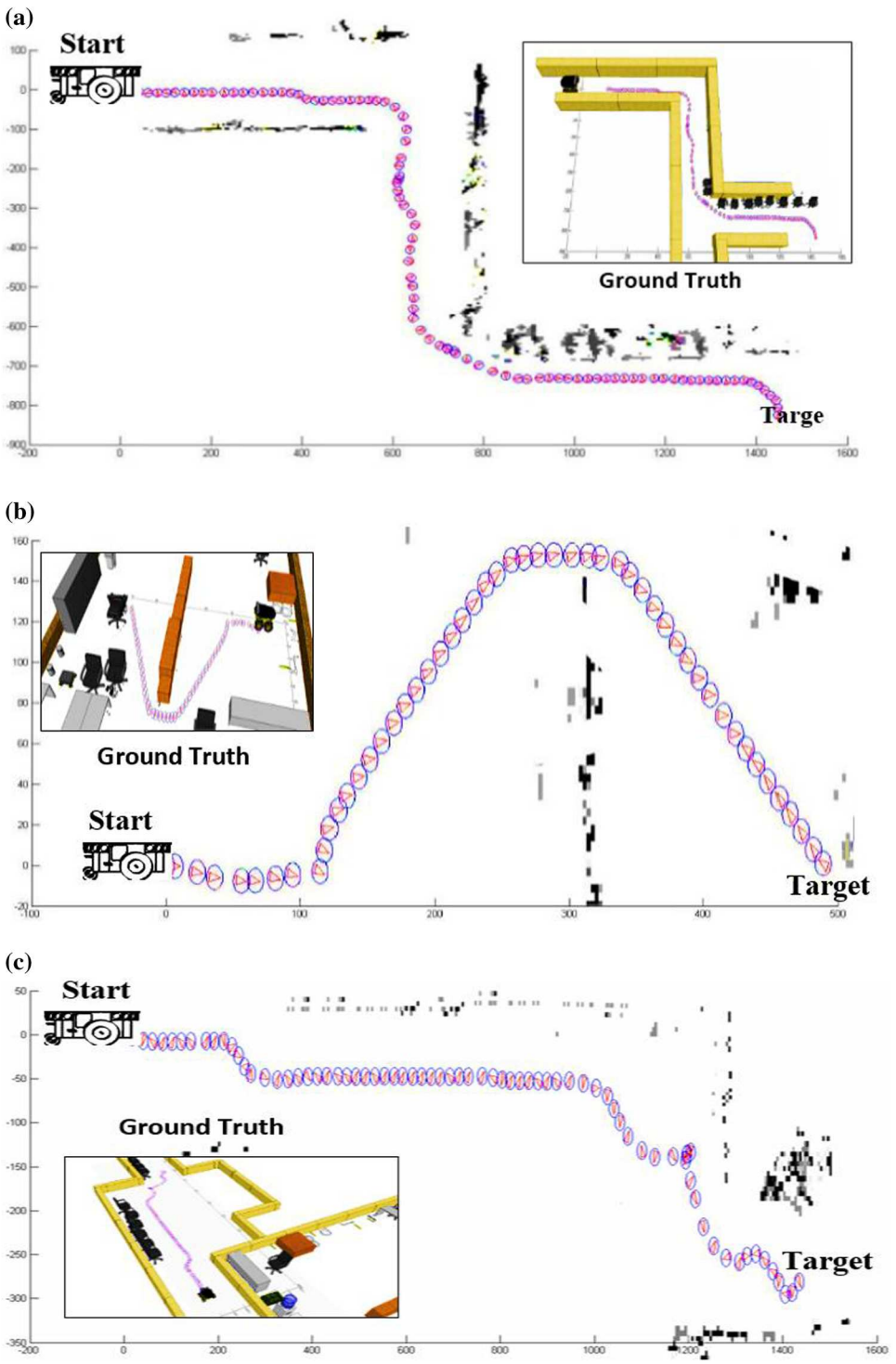


Fig. 8 Path-planning and trajectory-following outputs of ASNS in a corridor (**a**), inside room (**b**) and open indoor space (**c**)

dor floor. A low number of obstacles and open spaces usually require minimal path re-planning; thus, the average recorded speed for the robot was the highest in the experiment. Speed profiles for experiments conducted in various indoor environments are displayed for reference in Fig. 9.

7 Real-time execution analysis of ASNS

Important highlights listing the most time-consuming programming constructs for each module are recorded to analyze the execution time of the proposed system. The system is GPU enabled, and its real-time execution analysis is pre-

Table 7 Average system iteration time of path-planning and trajectory-following performance indicators for ASNS

Environment	Obstacle type	Average system iteration time (std. dev)
Tight spaces and small rooms	Low number of obstacles	1734 ms (241.7)
	Medium number of obstacles	1861 ms (412.2)
	Large number of obstacles	2079 ms (492.6)
Open spaces	Low number of obstacles	1321 ms (243.1)
	Medium number of obstacles	1455 ms (352.9)
	Large number of obstacles	1592 ms (377.2)
Corridor	Low number of obstacles	1648 ms (286.1)
	Medium number of obstacles	1690 ms (413.9)
	Large number of obstacles	1854 ms (441.7)

All listed experimental times are calculated by factoring out the speed-up via the GPU on board the robot

Table 8 Position accuracy of navigation performance with respect to ground truth

Scenario	Actual target position (meter)		Robot target position (meter)		Absolute error (meter)	
	x	*y*	*x*	*y*	x	y
Fig. 8-a	14	−8.5	13.95	−8.48	0.05	0.02
Fig. 8-b	5	0	5.02	0.02	0.02	0.02
Fig. 8-c	14	−3	13.97	2.97	0.03	0.03

Table 9 Path-planning and trajectory-following performance indicators for ASNS. All listed experimental times are calculated by factoring out the speed-up via the GPU on board the robot

Environment	Obstacle type	Average robot speed (std. dev)
Tight spaces and small rooms	Low number of obstacles	0.16 m/s (0.121)
	Medium number of obstacles	0.13 m/s (0.091)
	Large number of obstacles	0.15 m/s (0.049)
Open spaces	Low number of obstacles	0.46 m/s (0.096)
	Medium number of obstacles	0.28 m/s (0.055)
	Large number of obstacles	0.35 m/s (0.031)
Corridor	Low number of obstacles	0.44 m/s (0.067)
	Medium number of obstacles	0.33 m/s(0.042)

sented in Table 10. A snapshot of the robot speed profile outputs of ASNS for some results captured from some experiments are displayed in Fig. 9. The speed of the robot during the movement is measured from start point to target point in corridor "a", inside room "b", and open indoor space "c". The darker shades of blue indicate the high speeds. The lighter shades of blue denote lower speeds. The distance is in centimeters.

The specifications of the client-end and server-end machines and GPU system are provided below.

Client-end PC Specifications:

Processor: Intel Core i7
RAM: 16 GB
Hard Drive: 40 GB Solid-State Drive
Operating System: Windows Embedded Standard 7, 64-bit
Motherboard Specs:
Supports 3rd Gen. Intel® 22 nm CPUs and 2nd Gen. Intel®

Client-end GPU Specifications:
GPU: NVIDIA QUADRO K-4000
CUDA Parallel-Processing Cores: 768

Server-end PC Specifications:

Processor: Intel® Core™ i7 2630QM Processor
RAM: 12 GB DDR3 1333 MHz SDRAM
Operating System: Windows 7 Professional, 64-bit
Motherboard Specs: Intel® HM65 Express Chipset

8 Comparison with other works

As explained previously, the designed system was built with integrated various enhanced modules, algorithms and techniques built by us. Therefore, it is difficult to compare the whole system with another system. Therefore, we are going to compare some main modules of the designed system with another work.

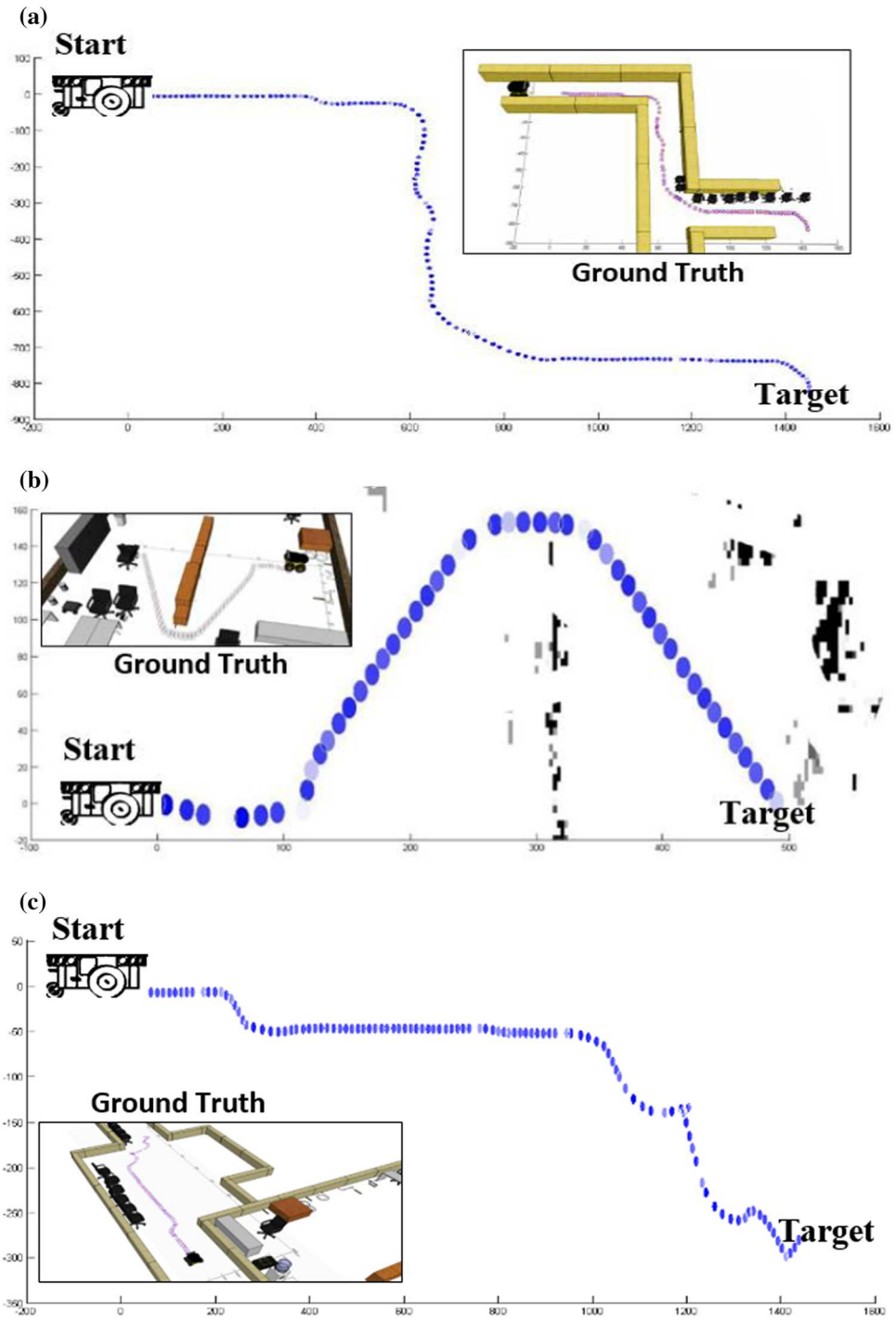


Fig. 9 Robot speed profile outputs of ASNS in a corridor (**a**), a small room (**b**) and open indoor space (**c**)

8.1 MMSVF

In Fig. 10, we compare the grid maps generated (filtered) by our MMSVF with those generated by Triclops SDK (Point Grey Research Inc.). Fig. 10a provides the grid maps generated by MMSVF, while Fig. 10b provides the grid maps generated by Triclops SDK. Comparing Figs. 10a and 9b, we can see that the grid map generated by Triclops SDK loses fidelity at the cost of noise removal, which indicates the high performance of the proposed filter.

As mentioned previously, we gather stereo point cloud sequences through the stereovision camera mounted on top of a mobile robot. These sequences are gathered while the robot is in motion, and the robot/camera pose is bundled with each observation. The observations are processed online by the proposed method in real time. The maximum speed

Table 10 The Real-time execution analysis of the system

	SOCM (std. dev)	ORWM (std. dev)	FLMC (std. dev)	IGM (std. dev)	MM (std. dev)	PPM (std. dev)
Individual runtime of modules	1047 ms (202.8)	719 ms (190.1)	306 ms (62.5)	59 ms (11.7)	175 ms (43.3)	85 ms (29.3)
Contribution to single iteration runtime during threaded execution (parallel processing)	1047 ms (202.8)	232 ms (74.2)	–	–	125 ms (27.2)	51 ms (11.5)
Contribution to single iteration runtime during both GPU-based and threaded execution	725 ms (111.2)	193 ms (69.4)	–	–	92 ms (21.6)	32 ms (8.2)

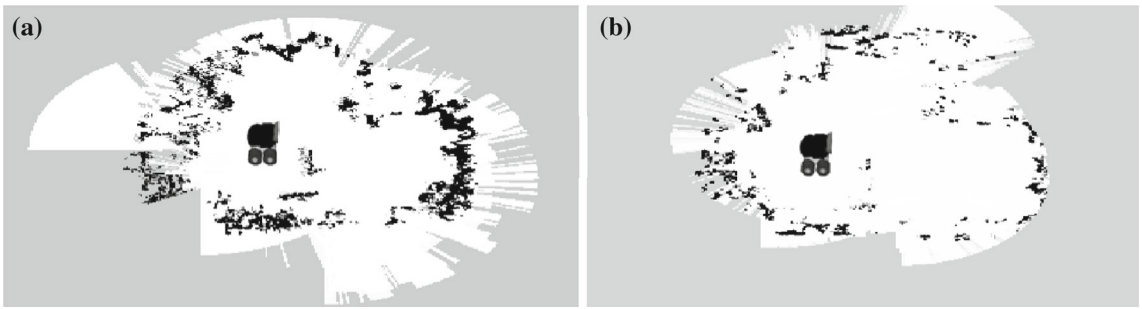


Fig. 10 Stochastic occupancy grid maps. *Left* mapping output from ICP-SLAM after application of proposed filter. The map contains minimal noise without the loss of fidelity. *Right* mapping out from ICP-SLAM after application of surface size, texture validation and back–forth filter by Triclops SDK, Point Grey Research. The map loses fidelity at the cost of noise removal. **a** Grid maps generated by MMSVF. **b** Grid maps generated by Triclops SDK

achieved by the robot during navigation is 0.47 m/s. For comparison, the ground truth for the arena is also displayed in Fig. 11.

We compared the depth estimation accuracy of the proposed MMSVF module with the work of Yoon et al. [46]. The comparison results are shown in Table 11, from far 3 m to near 0.5 m. The proposed MMSVF module shows better results than the compared one [46]. The maximum error of the proposed MMSVF was 0.04 m while in the compared method was 0.2 m.

Fig. 11 Ground truth for the occupancy grid map

8.2 IGM

The results of the IGM module are compiled with a shape that is comparable in the work of Aniket and Kuipers [47], where number of frames required for detection of an obstacle serves as a criterion to judge the performance of stereo-mapping algorithm. This compilation is shown in Table 12. The number of required frames was only recorded for obstacles located between 2.5 and 3.5 m from the camera.

8.3 AFDS

An accuracy comparison of the ground truth of the measured obstacle of the proposed method AFDS on two sensors (bumblebee and kinect sensor) is shown in Table 13. Twenty experiments were conducted with varying obstacle configurations for different (x_{res}, y_{res}), where the proposed method was executed for each sensor separately. The maxi-

Table 11 Depth estimation accuracy comparison

	Proposed MMSVF		Compared method [46]	
Real depth (m)	Estimated depth (m)	Errors (m)	Estimated depth (m)	Errors (m)
3	2.97	0.03	2.94	0.06
2.5	2.52	0.02	2.47	0.03
2	2.04	0.04	2.05	0.05
1.8	1.79	0.01	1.87	0.07
1.7	1.71	0.01	1.79	0.09
1.5	1.51	0.01	1.60	0.10
1.3	1.31	0.01	1.47	0.17
1	1.02	0.02	1.20	0.20
0.5	0.49	0.01	0.68	0.18

Table 12 IGM frames comparison

Nature of obstacle	Number of frames required (proposed strategy)	Number of frames required ([46])
Textured (more than 50 % area detected)	4	7
Non-textured (more than 50 % area detected)	5	8
Textured (more than 90 % area detected)	7	9
Non-textured (more than 90 % area detected)	21	32

Table 13 AFDS accuracy

x_{res}, y_{res}	Obstacle distance from sensor	Average error in maximum detected height	
		<3 m	>3 m<5.5 m
3, 2	Kinect sensor	± 0.51 cm	± 5.31 cm
	Bumblebee camera	± 0.39 cm	± 2.49 cm
4, 3	Kinect sensor	± 0.44 cm	± 4.97 cm
	Bumblebee camera	± 0.31 cm	± 2.21 cm
8, 6	Kinect sensor	± 0.44 cm	± 4.88 cm
	Bumblebee camera	± 0.32 cm	± 2.10 cm
12, 9	Kinect sensor	± 0.40 cm	± 4.74 cm
	Bumblebee camera	± 0.29 cm	± 1.99 cm

Table 14 The real-time execution analysis of the AFDS

Sequence name	Observations (frames)	Overall system runtime (s)	Algorithm runtime (s)
FR1 360	745	427	383
FR1 Room	1332	795	568
FR1 Floor	1214	742	485

mum height of each obstacle was measured manually and served as ground truth. The errors between the detected maximum heights of the grid map vertices and the ground truth were calculated. These average errors are shown in Table 13.

A real-time performance comparison of the AFDS was analyzed using a well-documented RGB-D SLAM dataset [48] with different sequence name and observations, as illustrated in Table 14. The *AFDS* requires on average approximately 0.47 s of processing time per observation.

9 Conclusion

In this paper, an intelligent motion system for mobile robots is designed and implemented using a multi-threaded and client–server based architecture. The system is able to achieve an autonomous navigation within an unstructured dynamic indoor environment. The system has many characteristics: stereovision-based navigation within an unknown dynamic indoor environment, adaptability to indoor lighting conditions via stereo-matching parameters and point cloud filtering

parameter tuning, the capability to utilize previously built maps for navigation, path-planning and localization tuning via parameters, a laser measurement sensor-assisted system for emergency braking and minor path adjustment, and the capability to move through multiple waypoints while performing fully featured navigation. We showed that the main modules of our system had better performance than the similar modules in the literatures. A major contribution of this paper is that we presented the performance of the proposed ASNA system, integrating all modules, in different environments using different parameters and architectures.

Acknowledgments The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through Research Group No. RG-1437-018.

References

1. Lategahn H, Derendarz W, Graf T, Kitt B, Effertz J (2010) Occupancy grid computation from dense stereo and sparse structure and motion points for automotive applications. In: Intelligent vehicles symposium (IV), 2010 IEEE, pp 819–824
2. Shi C, Wang G, Yin X, Pei X, He B, Lin X (2012) High-accuracy stereo matching based on adaptive ground control points. Image Processing, IEEE Transactions on 24(4):1412–1423
3. Mei X, Sun X, Zhou M, Jiao S, Wang H, Zhang X (2011) On building an accurate stereo matching system on graphics hardware. In: IEEE international conference on computer vision workshops (ICCV Workshops), pp 467–474
4. Ambrosch K, Kubinger W (2010) Accurate hardware-based stereo vision. Comput Vis Image Underst 114(11):1303–1316
5. Lemaire T, Berger C, Jung I-K, Lacroix S (2007) Vision-based slam: stereo and monocular approaches. Int J Comput Vis 74(3):343–364
6. Rusu RB, Blodow N, Beetz M (2009) Fast point feature histograms (FPFH) for 3D registration. In: IEEE international conference on robotics and automation, 2009. ICRA'09, pp 3212–3217
7. Steder B, Rusu RB, Konolige K, Burgard W (2011) Point feature extraction on 3D range scans taking into account object boundaries. In: IEEE international conference on robotics and automation (ICRA), pp 2601–2608
8. Johnson AE, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans Pattern Anal Mach Intell 21(5):433–449
9. Stiene S, Lingemann K, Nuchter A, Hertzberg J (2006) Contour-based object detection in range images. In: Third international symposium on 3D data processing, visualization, and transmission, IEEE, pp 168–175
10. Ledwich L, Williams S (2004) Reduced SIFT features for image retrieval and indoor localisation. In: Australian conference on robotics and automation, 2004. Citeseer
11. Schleicher D, Bergasa LM, Barea R, Lopez E, Ocaa M, Nuevo J, Fernandez P (2007) Real-time stereo visual slam in large-scale environments based on sift fingerprints. In: IEEE international symposium on intelligent signal processing, 2007. WISP 2007, pp 1–6
12. Sinha U (2010) Sift: scale invariant feature transform. AI Shack 14, Harvard
13. Bais A, Sablatnig R, Gu J, Khawaja YM, Usman M, Hasan GM, Iqbal MT (2008) Stereo vision based self-localization of autonomous mobile robots. In: Sommer G, Klette R (eds) Robot Vision. Springer, Berlin, pp 367–380
14. Ghayalod MP, Hall EL (1992) Intelligent robot control using omnidirectional vision. In: Applications in optical science and engineering, 1992. International society for optics and photonics, pp 573–584
15. Inaba M, Kanehiro F, Kagami S, Inoue H (1995) Vision-equipped apelike robot based on the remote-brained approach. In: 1995 IEEE international conference on robotics and automation. 1995. Proceedings, pp 2193–2198
16. Matthies L, Shafer S (1987) Error modeling in stereo navigation. J IEEE Robot Autom 3(3):239–248
17. Fazl-Ersi E, Tsotsos JK (2009) Region classification for robust floor detection in indoor environments. In: Kamel M, Campilho A (eds) Image analysis and recognition. Springer, Berlin, pp 717–726
18. Murray D, Jennings C (1997) Stereo vision based mapping and navigation for mobile robots. In: 1997 IEEE international conference on robotics and automation. Proceedings 1997, pp 1694–1699
19. Elfes A (1989) Using occupancy grids for mobile robot perception and navigation. Computer 22(6):46–57
20. Kuhn A, Hirschmüller H, Mayer H (2013) Multi-resolution range data fusion for multi-view stereo reconstruction. In: Weickert J, Hein M, Schiele B (eds) Pattern Recognition. Springer, Berlin, pp 41–50
21. Milella A, Reina G, Foglia MM (2013) A multi-baseline stereo system for scene segmentation in natural environments. In: 2013 IEEE international conference on technologies for practical robot applications (TePRA), pp 1–6
22. Gallup D, Frahm J-M, Mordohai P, Pollefeys M (2008) Variable baseline/resolution stereo. In: IEEE conference on computer vision and pattern recognition, 2008. CVPR 2008, pp 1–8
23. Wang H, Xu J, Guzman JI, Jarvis RA, Goh T, Chan CW (2001) Real time obstacle detection for AGV navigation using multi-baseline stereo. In: Rus D, Singh S (eds) Experimental robotics VII. Springer, Berlin, pp 561–568
24. Pahlavan K, Eklundh J-O (1992) A head-eye system-analysis and design. CVGIP Image Underst 56(1):41–56
25. Samson E, Laurendeau D, Parizeau M, Comtois S, Allan J-F, Gosselin C (2006) The agile stereo pair for active vision. Mach Vis Appl 17(1):32–50
26. Cindy X, Collange F, Jurie F, Martinet P (2001) Object tracking with a pan-tilt-zoom camera: application to car driving assistance. In: IEEE International Conference on Robotics and automation. Proceedings 2001, ICRA, 2001 IEEE, pp 1653-1658
27. Milios E, Jenkin M (1993) Torsional eye movements. In: IEEE International Conference on Act Robot Vis Camera Heads Model Based Navig React Control, vol 6, p 51
28. Nakagawa M, Adachi E, Takase R, Okamura Y, Kawai Y, Yoshimi T, Tomita F (2008) Gaze tracking control using an active stereo camera. Int Arch Photogramm Remote Sens Spat Inf Sci 37(Part B3b):387–392
29. Kühnlenz K, Lidoris G, Wollherr D, Buss M (2007) On foveated gaze control and combined gaze and locomotion planning. INTECH Open Access Publisher
30. Stachniss C, Grisetti G, Burgard W (2005) Information gain-based exploration using rao-blackwellized particle filters. In: Robotics science and systems, vol 2, pp 65–72
31. Bourgault F, Makarenko AA, Williams SB, Grocholsky B, Durrant-Whyte HF (2002) Information based adaptive robotic exploration. In: IEEE/RSJ international conference on intelligent robots and systems, pp 540–545
32. Rashid R, Elamvazuthi I, Begam M, Arrofiq M (2010) Differential drive wheeled mobile robot (WMR) control using fuzzy logic techniques. In: 2010 Fourth Asia International Conference on mathematical/analytical modelling and computer simulation (AMS), pp 51–55

33. Faisal M, Hedjar R, Al Sulaiman M, Al-Mutib K (2013) Fuzzy logic navigation and obstacle avoidance by a mobile robot in an unknown dynamic environment. Int J Adv Robot Syst 10
34. Raudonis V, Maskeliunas R (2011) Trajectory based fuzzy controller for indoor navigation. In: 2011 IEEE 12th International Symposium on computational Intelligence and Informatics (CINTI), pp 69–72
35. Li T-H, Chang S-J, Tong W (2004) Fuzzy target tracking control of autonomous mobile robots by using infrared sensors. IEEE Trans Fuzzy Syst 12(4):491–501
36. Narvydas G, Simutis R, Raudonis V (2007) Autonomous mobile robot control using fuzzy logic and genetic algorithm. In: 4th IEEE Workshop on intelligent data acquisition and advanced computing systems: technology and applications, 2007. IDAACS 2007, pp 460–464
37. Cupertino F, Giordano V, Naso D, Delfine L (2006) Fuzzy control of a mobile robot. IEEE Robot Autom Mag 13(4):74–81
38. Fua P (1991) Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In: International joint conference on artificial intelligence (IJCAI)
39. Murray D, Little JJ (2000) Using real-time stereo vision for mobile robot navigation. Auton Robot 8(2):161–171
40. Głąbowski M, Musznicki B, Nowak P, Zwierzykowski P (2012) Shortest path problem solving based on ant colony optimization metaheuristic. Image Process Commun 17(1–2):7–17
41. Thrun S, Burgard W, Fox D (2005) Probabilistic robotics, vol 1. MIT press, Cambridge
42. Khalid Almutib ME, Mansour Alsulaiman, Hedjar Ramdane and Ebrahim Mattar (2014) Reliable multi-baseline stereovision filter for navigation in unknown indoor environments. In: Paper presented at the 29th international conference on computers and their applications (CATA), Las Vegas, Nevada, USA
43. Emaduddin M, AlMutib K, AlSulaiman M, Hedjar R, Mattar E (2012) Accurate floor detection and segmentation for indoor navigation using RGB+ D and stereo cameras. In: Proceedings of the 2010 international conference on image processing, computer vision, pattern recognition (2012)
44. Faisal M, A-M K, Hedjar R, Mathkour H, Alsulaiman M, Mattar E (2014) Behavior based mobile for mobile robots navigation and obstacle avoidance. Int J Comput Commun 8
45. Khalid Almutib ME, Mansour Alsulaiman, Hedjar Ramdane, Ebrahim Mattar (2014) Smart stereovision based gaze control for navigation in low-feature unknown indoor environments. In: 5th intelligent systems, modeling and simulation (ISMS), Langkawi, Malaysia
46. Yoon H-J, Hwang Y-C, Cha E-Y (2010) Real-time container position estimation method using stereo vision for container autolanding system. In: 2010 International conference on control automation and systems (ICCAS), pp 872–876
47. Murarka A, Kuipers B (2009) A stereo vision based mapping algorithm for detecting inclines, drop-offs, and obstacles for safe local navigation. In: IEEE/RSJ international conference on intelligent robots and systems, 2009. IROS 2009, pp 1646–1653
48. Sturm J, Magnenat S, Engelhard N, Pomerleau F, Colas F, Burgard W, Cremers D, Siegwart R (2011) Towards a benchmark for RGB-D SLAM evaluation. In: Proceedings of the RGB-D workshop on advanced reasoning with depth cameras at robotics: science and systems conference (RSS), Los Angeles, USA, p 3