Robotics and Autonomous Systems 70 (2015) 166–180

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Thorough robot navigation based on SVM local planning

Konstantinos Charalampous *[,1], Ioannis Kostavelis [1], Antonios Gasteratos [1]

Laboratory of Robotics and Automation, Production and Management Engineering Department, Democritus University of Thrace, Vas. Sophias 12, GR-671 00 Xanthi, Greece

HIGHLIGHTS

- Autonomous robot path planning.
- SVM based local planning.
- Global map formation.

ARTICLE INFO

Article history:
Received 6 March 2014
Received in revised form
21 October 2014
Accepted 16 February 2015
Available online 23 February 2015

Keywords:
Path planning
SVM
Robot navigation
Point cloud

ABSTRACT

A prerequisite for autonomous robot navigation is the extraction of a path that is both efficient and safe in terms of collision. Towards this end, the paper in hand presents a novel local path planning method, incorporating the *support vector machines* (SVM) theory. The original SVM based module exploits a 2D map of points which are considered to be obstacles, so as to culminate in a collision free path. A unique attribute of the proposed SVM based local path planning algorithm is that it considers the consecutive positions of the global path trajectory, the embodiment of the robot and clusters the obstacles accordingly. Thus, the derived trajectory is a physically constrained path inasmuch as it considers the maximum margin notion of the SVM theory. Instead of providing a purely theoretical approach for local planning assessed using only artificial data, we integrate our local planner into an autonomous navigation system which is evaluated in real-world scenarios in order to show its efficacy. The latter framework firstly constructs a global 3D metric map of the perceived environment and then it converts it into a 2D map upon which a global path planner unrolls. The global map grows incrementally, by registering the collected point clouds over the robot's route towards a goal position. Moreover, the navigation is supported by an obstacle detection strategy based on v-disparity images. The system – and, consequently, the presented local path planner – was evaluated in long range outdoors scenarios, navigating successfully within congestive environments.

1. Introduction

Autonomous robot navigation is a research topic that has notably attracted attention over the last decades, due to the fact that robots establish more and more their presence in a variety of applications, spanning from industrial to space exploration ones, as well as our daily lives. Mobile robots intended to be employed in household environments should be capable of attaining a range of applications such as cleaning a room, and fetching tasks. With the aim to competently achieve this, safe navigation and continuous collision avoidance is of great demand. Thereby, the respective systems should be in position to deal with static obstacles, i.e. of known position and volume within their surroundings, as well as with dynamic ones that might emerge while navigating. Additionally, the planning of such paths should be time efficient, notwithstanding the achievement of this objective is obstructed especially in cases where the *a priori* knowledge of the environment is limited and the already extracted route needs frequent revisions. Given a geometric map of the explored environment, the autonomous robot navigation comprises two separate indispensable scientific areas, viz. path planning and robot localization. The aforementioned areas have been studied thoroughly, yielding a variety of solutions for each one individually.

Robot localization deals with the determination of a robot's pose (i.e. location and orientation) in accordance with a global

* Corresponding author.
E-mail addresses: kchara@pme.duth.gr (K. Charalampous), gkostave@pme.duth.gr (I. Kostavelis), agaster@pme.duth.gr (A. Gasteratos).
[1] Tel.: +30 2541 0 79330. URL: http://robotics.pme.duth.gr.

http://dx.doi.org/10.1016/j.robot.2015.02.010
0921-8890/

coordinate system. Two well known families of algorithms that deal with this problem are based on Kalman filtering and on *Monte Carlo localization* (MCL). A representative method that relies on Kalman filter is the extended Kalman filter [1,2], which in a nutshell models both the robot's sensor measurements and the state transition as Markov processes that are further described by non-linear functions. The key-points of this method are: the linear approximation of the functions through Taylor expansion and the mixture of the sensorial and odometry data using the Kalman filter [3]. MCL based approaches rely upon particle filters and constitute a frequent choice to confront with the localization problem. Such methods express robot's posterior belief according to a set of weighted hypotheses (particles) which approximate the posterior [4,5]. However, they also exhibit several disadvantages, such as the computational complexity, i.e. the required number of particles is directly burdened with the scale of the explored environment. Several revisions of the MCL were considered with the aim to improve its performance, such as *augmented* MCL [2], *mixture* MCL [6] and *self-adaptive* MCL [7].

Regarding the robot path planning issue, two bunches of methods are distinguished, namely the global and the local ones, which act complementarily. *Global path planning* (GPP) operates in high level, performs long term planning, seeks for the shortest path to a goal according to certain criteria and avoids *dead end* situations. Besides, *local path planning* (LPP) focuses on smaller obstacles and derives a feasible, continuous, obstacle free trajectory in a time efficient fashion. Both problems are thoroughly studied individually [8–10], yet the recent notion is the proposition of hybrid methodologies that provide both global and local solutions [11–14]. Rashid et al. [11] presented a path planning method based on Bresenham algorithms, which are utilized for the trajectory formation, and the global path derives through graph construction procedures and searching tactics. The work in [12] proposed an online path planning algorithm that relies on the network simplex methodology, which was integrated with the virtual vehicle approach to form smooth feasible trajectories. In [13] the authors modified the frontier-based exploration technique to a path planning algorithm, which was further blended with the potential field method, deducing a navigation framework that avoids dynamic obstacles. The authors in [14] relied on the concepts of adaptive dimensionality reduction and developed a single graph for local and global path planning that guarantees completeness while operating in an online fashion.

Another family of algorithms that are used widely for this task is the one that relies on Voronoi graphs. The authors in [15] propose a framework that combines the fingerprint concept along with uncertainty for navigation in unknown environment. More precisely, they generate fingerprints from the respective sensors and the noise is modeled as uncertainty on the derived features. A dynamic programming approach is utilized to match these features, while the navigation part uses a Partially Observable Markov Decision Process model. The work in [16] introduced a SLAM method that exploits the Generalized Voronoi Graph (GVG) to form the topological map. The edges and the nodes of the respective GVG graph facilitate robot's navigation in an unknown environment, while the corresponding graph structure is used for localization. A similar approach to GVG, namely the Extended Voronoi Graph (EVG) [17], is capable of providing smooth transitions from corridors to wider places. The formation of the EVG does not require any prior knowledge yet relies on an accurate place detection routine. Moreover, there are several issues that arise while navigating and several extensions have been proposed in order to be avoided. Voronoi graphs have proven to operate appropriately in places such as corridors, yet the robot's transitioning to a non-corridor place disuses the graph. A suitable solution to this problem is coastal navigation which demands the preservation of an obstacle within the field of view permanently. Once there are not any visible obstacles, a fact that occurs often in robot navigation, then the graph is no longer able to be created from the respective sensorial information and the robot cannot be self localized. Another issue that is yielded is the sensitivity to noise introduced from sensors which leads to spurious formation of places. Moreover, a known weakness of Voronoi graph-based methods is the removal or isolation of places at "L" intersections, which of course are significant with respect to topological deduction.

Last, Although Potential Field Methods (PFM) which are also widely used for autonomous robot navigation have several drawbacks and limitations [18]. A problem that commonly occurs in such systems is the local minima one. Such situations occur and the robot is lead to a dead end, for example in a U-shaped obstacle, reasoning the existence of several works that attempt to solve the latter issue mainly using heuristic functions. Another issue that arises in PFM is the incapability to traverse through close obstacles, although the robot's embodiment is sufficient for passing through. This holds due to the fact that the repulsive forces of the obstacles are summed pointing away from the space that exists between them. Another important phenomenon that is present in PFM is the robot's unstable cruising caused by the presence of obstacles. Last, oscillations in a robot's trajectory also can take place when the latter traverses in a narrow passage and repulsive forces from opposite sides act upon it.

Autonomous robot navigation comprises the integration of the aforementioned areas along with the respective sensors that provide input to the algorithms. The literature cites several works exhibiting results from such systems. Petereit et al. [19] presented an extension of hybrid A*, suitable for path planning in unstructured outdoors environments that enables the formation of feasible paths, navigable by the robot. In particular, it concerns the insertion of successive endpoints; by adjusting the pose when the robot reaches such a point, the reachability to the following one is secured and so on. The terrain characteristics were also considered, which were expressed as a weight to each cell upon the occupancy grid. Moreover, heuristic functions enabled wall following capabilities and waypoints that lie outside its local map. The modifications and additions to the heuristic functions as well as to the cost ones, led to a system that prefers navigating on-road and deals with obstacles that surpass the dimension of the map. Wurm et al. [20] presented an improved autonomous navigation system that detects vegetation and avoids it. The methodology exploits the measured range, the incidence angle and the remission values of a laser scanner to form feature vectors, i.e. samples. The latter is classified using SVM, the training of which was accomplished off-line using a self-supervised scheme, instead of manually labeling the data. By considering the classification of the terrain for the path planning, the overall navigation system produces a secure trajectory over the streets.

The ongoing encapsulation of point clouds and machine learning techniques during the last years yielded state-of-the art algorithms in robotics [21,22]. Regarding path planning, the work presented in [23] is a framework that incorporates both hardware and software to constantly form a 3D point cloud. The framework is endowed with a pose estimation algorithm fusing IMU data and wheel odometry. Furthermore, the path planning procedure relies upon a dynamic programming process and it is applied on the calculated dense metric map. In [24] the authors proposed a system that improves navigation performance based on self-supervised learning. This planning and mapping structure incorporates a low-level detection scheme for very distant obstacles, along with a deliberative long-range and hyperbolic-polar planner. In [25] a computationally competent method for the traversability estimation of the terrain, based on SVM classification, applied on elaborated stereo pair images. The scenes detected to be traversable

are further processed to estimate the distribution of obstacle likelihoods in front of the robot, taking into account its embodiment.

Besides, the pure geometrical attributes of the SVMs can be used to address the path planning problem. Thus, by following a strictly theoretical approximation Miura [26] presented a conceptual work that involves the geometrical principle of estimating a separation hyperplane using maxima distances upon which the classifier is built to plot robot's intended route. In particular, this work comprises an iterative process, exploiting several different patterns seeking for a feasible path. The authors in [27,28] used SVM to cope with the *k*-nearest neighbor classifier for path planning in unknown environments. However, the aforementioned papers evaluated their techniques in a simulated noise free environment. The authors in [29] utilized a basic subdivision method for local maps, they compute the respective boundary points and then they calculate a path by means of an SVM. Nonetheless, there are several limitations concerning the parameters of the SVM, such as the kernel selection, as well as the simplification that the environment comprises non-convex obstacles. In spite of the fact that these methods competently solve the path planning problem, they all operate in an straightforward fashion concealing the real potentialities that such a system is apt to retain.

The paper in hand proposes an *SVM local planner* (SVMLP) to determine the path in an unknown environment. In contrast to previously proposed works, where an iterative scheme takes place until a criterion is met, SVMLP is trained only once. The combination of GPPs that seek for the shortest path along with the proposed SVMLP that derives such secure trajectory retains both optimality and safeness during the robot's navigation. The novel LPP administers the continuous trajectory among the successive points of the GPP. Moreover, the suggested LPP utilizes machine learning techniques, such as clustering and SVM to deduce a safe path along obstacles. The integrated system in which the SVMLP was evaluated is capable of generating 3D point clouds of the robot's surroundings during its travel in an unexplored environment. The point clouds are further processed to extract a 2D map, which is fed as an input to the GPP. As the robot travels from a starting to an intended position, the visual odometry module provides its respective localization. Additionally, obstacles that due to size are indistinguishable on the 2D map are engaged by the obstacle detection routine. To the best of our knowledge, the proposed work introduces here, for the first time, a long range autonomous robot navigation framework which utilizes SVM for the local path planning in a closed loop fashion. Moreover, the performance of the proposed SMVLP is evaluated on an integrated robotic system providing competent navigation capacities.

2. Proposed method

This section covers the entire framework of the system in which the local planner is incorporated. Starting from the distinctive modules – viz. the point cloud registration, the extraction of the 2D map, the GPP, the visual odometry and the SVMLP – the section concludes describing their integration into one concrete system. Although the novelty of the proposed work stays with the local planner, for completeness purposes it is presented here within the context of an intact autonomous navigation system that makes use of SVMLP as a local planner in real outdoors scenarios. Therefore, this section outlines the full-scale system and briefly describes the other components.

2.1. Global map generation

The registration of 360° point clouds employs a two-fold procedure. First, according to the robot's localization (see Section 2.3.1) a transformation matrix is estimated, which is used for a rough alignment between the existing map and the newly acquired point cloud. This step ensures not only the closeness between the pose of the new full circle point cloud and the real transformation, but also speeds up the duration of the upcoming step. This is a refinement one, incorporating the *fast point feature histograms* (FPFH) [30] which are multi-dimensional features describing the geometry of a point belonging to a 3D point cloud. A thresholding procedure subsequently rejects correspondences wherever the distance between them is excessive. The latter occurs because there exist several erroneous correspondences between the point clouds which can affect in a negative fashion the estimation of the transformation. The rejection may be accomplished via either a RANSAC based solution or a trimming procedure, retaining a percentage of correspondences. In this work we utilize a correspondence routine that relies on thresholding the Euclidean distances between the correspondences of the two point clouds. Let us assume that the robot acquires two specific point clouds ${}^{t}\mathbf{P}'$ and ${}^{t+1}\mathbf{P}'$ at times t and $t+1$, respectively. The robot's specific motion is described by a rotation ${}^{t}_{t+1}\mathbf{R}$ and translation matrix ${}^{t}_{t+1}\mathbf{T}$ and is calculated via the transformation among the FPFH correspondences resulting from the two point clouds as follows: ${}^{t}\mathbf{P}_{FPFH} = {}^{t}_{t+1}\mathbf{R} \cdot \mathbf{P}_{FPFH} + {}^{t}_{t+1}\mathbf{T}$. Then, the computation of the transformation matrix is performed by means of *singular value decomposition* (SVD). This transformation matrix is also applied on the new point cloud, improving further the respective registration. The *iterative closest point* (ICP) algorithm concludes this step by undertaking the fine tuning of the registration. Since the point clouds have already been registered, the ICP requires a small number of iterations and, as a result, the overall computation duration is significantly reduced. The particular algorithm employed here is the *point-to-plane* ICP [31], resulting to a transformation $[{}^{t}_{t+1}\mathbf{R}_{ICP}, {}^{t}_{t+1}\mathbf{T}_{ICP}]$, which is combined with the estimation from the FPFH registration, i.e. ${}^{t}_{t+1}\mathbf{R}_{ref} = {}^{t}_{t+1}\mathbf{R} \cdot \mathbf{R}_{ICP}$ and ${}^{t}_{t+1}\mathbf{T}_{ref} = {}^{t}_{t+1}\mathbf{T} + \mathbf{T}_{ICP}$. Fig. 1(a) illustrates the point cloud registration by applying only the transformation deduced by the localization component, whilst Fig. 1(b) depicts the registered point clouds, whence – by following a rough alignment step – a refinement one occurs producing more accurate and consistent results. That is, the aforementioned step invokes essentially on the 3D point cloud registration. The formation of the 2D global map $\mathcal{M}_2$ of the environment the robot navigates within, is based on the registered point clouds in five processing steps. First, all the points inside the bounding box of the robot are removed, since the robot itself should not be considered as an obstacle. The second step involves the removal of the ground, by means of the RANSAC algorithm [32], which determines the points belonging to the dominant plane perpendicular to the z-axis. Afterward, the points higher than the robot are also removed to avoid their being marked as obstacle points, owing to the fact that it is impossible for the robot to collide with. The 3D map is further filtered to remove duplicated points by applying a voxel grid filter with leaf size equal to 1 cm. Last, the remaining points of the aforementioned four steps are top-down projected, forming the global 2D map. The intermediate steps that derive the $\mathcal{M}_2$ map are illustrated in Fig. 2.

2.2. Global path planning

GPP is an essential part of autonomous robot navigation since it provides the path towards the ultimate goal position. Global path planners exploit *a priori* collected data from a variety of sensors, express the acquired information properly in the configuration space and seek for the optimal path according to the respective search algorithm. Moreover, in many robotics applications they are coupled with a local navigation system. The GPP undertakes the task of extracting an optimal discrete path, while the local one deals with the continuous space and the avoidance of small obstacles. There is a great variety of methodologies dealing with GPP, mainly due to lack of a generic definition of what an optimal

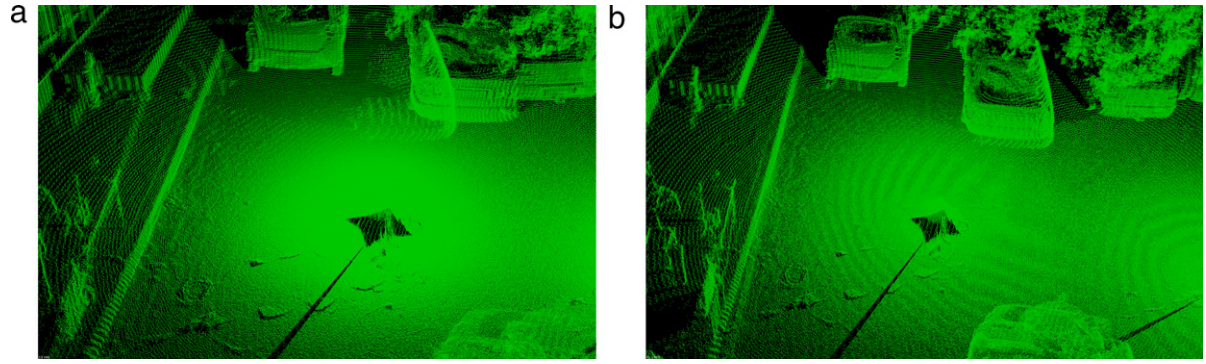


Fig. 1. (a) Initial point cloud registration using solely the system's localization component (rough estimation); (b) registered point clouds, where after the initial registration, the refinement step is performed using the correspondences of the FPFH features and the ICP algorithm. Note that the refined transformation is also applied on the robot's localization.

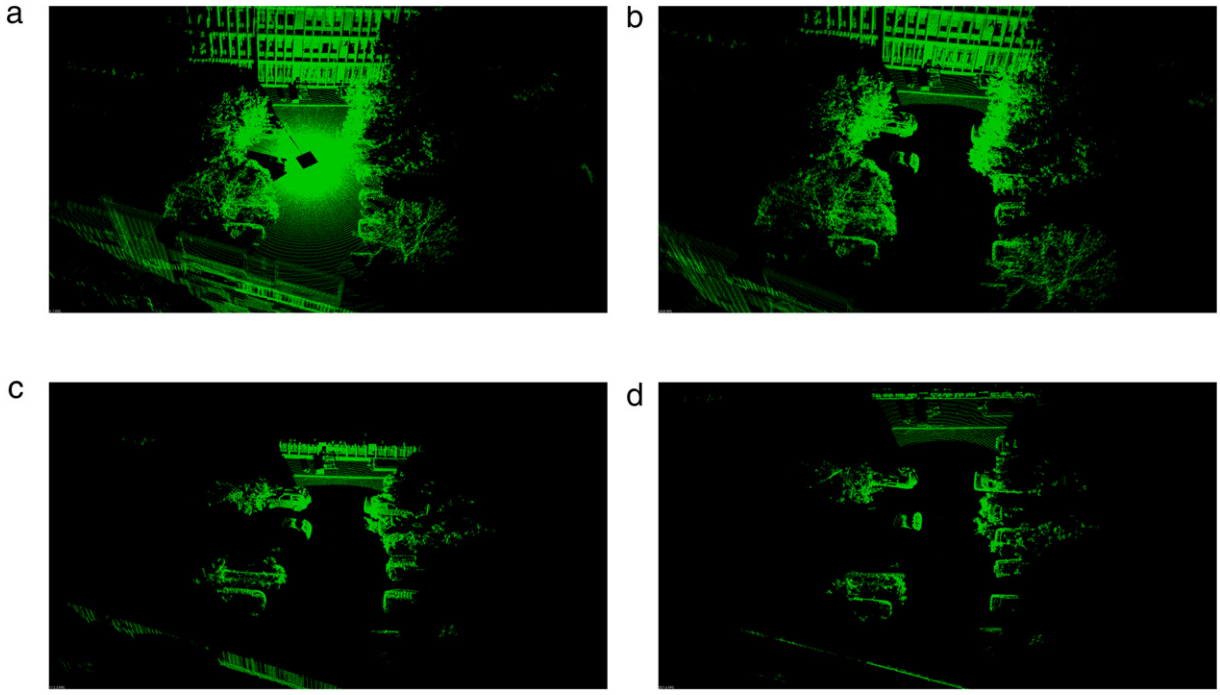


Fig. 2. (a) The initial point cloud, where only the points inside the bounding box of the robot have been removed; (b) the point cloud after the ground extraction; (c) the point cloud in which points higher than the robot are subtracted and (d) the final 2D global map after the filtering and the top down projection.

path is, which rather depends on the task to be accomplished and the conditions set. Nevertheless, in order to plan locally and, thus, evaluate the SVMLP, a global planning needs to be performed first. The SVMLP being a generic local planner may cooperate with the vast majority of the GPPs. The autonomous navigation system here exploits the D* Lite global path planner [33], which is a fast path planning and re-planning algorithm suitable for goal-directed robot navigation in partially known or unknown terrain. D* Lite constitutes an extension of Lifelong Planning A* (LPA*) [34], and it is capable of re-planning a new shortest route from its current position to the given goal when new obstacles appear.

2.3. Navigation in continuous space

2.3.1. Robot localization

Considering that the robot has to follow a certain list of points towards a target location, according to the global path planner, specific commands need to be passed to the robot's motor wheels. However, due to the drift errors when operating in slippery terrain, the robot rarely passes from the exact points. Therefore, we utilize visual odometry (VO) to accurately estimate the robot's displacement, relative to a starting location, exploiting solely stereo vision [35]. Particularly, among two consecutive stereo pairs, the most prominent features are detected and matched on the corresponding left images. The respective disparity maps are then computed and the matched features are registered with their disparity values and the estimated 3D coordinates. The erroneously matched landmarks are filtered out and the retained inliers are utilized by the motion estimation module to compute the optimal pose transformation of the robot among the consecutive movements. The utilized VO algorithm is the one described in [35] and extensively tested both on simulated and real world long range outdoors scenarios in [36]. The proposed algorithm is designed to operate in 100 m route with localization accuracy better than 2% during the entire route. Simultaneously, the orientation accuracy was retained as low as 5° for the entire route. Note that the localization accuracy is also refined by considering the transformations resulted during the registration of the point clouds, which is also adopted to further correct the robot's motion estimations.

The proposed navigation algorithm seeks for new obstacles at each motion estimation step (Fig. 3). The latter is achieved

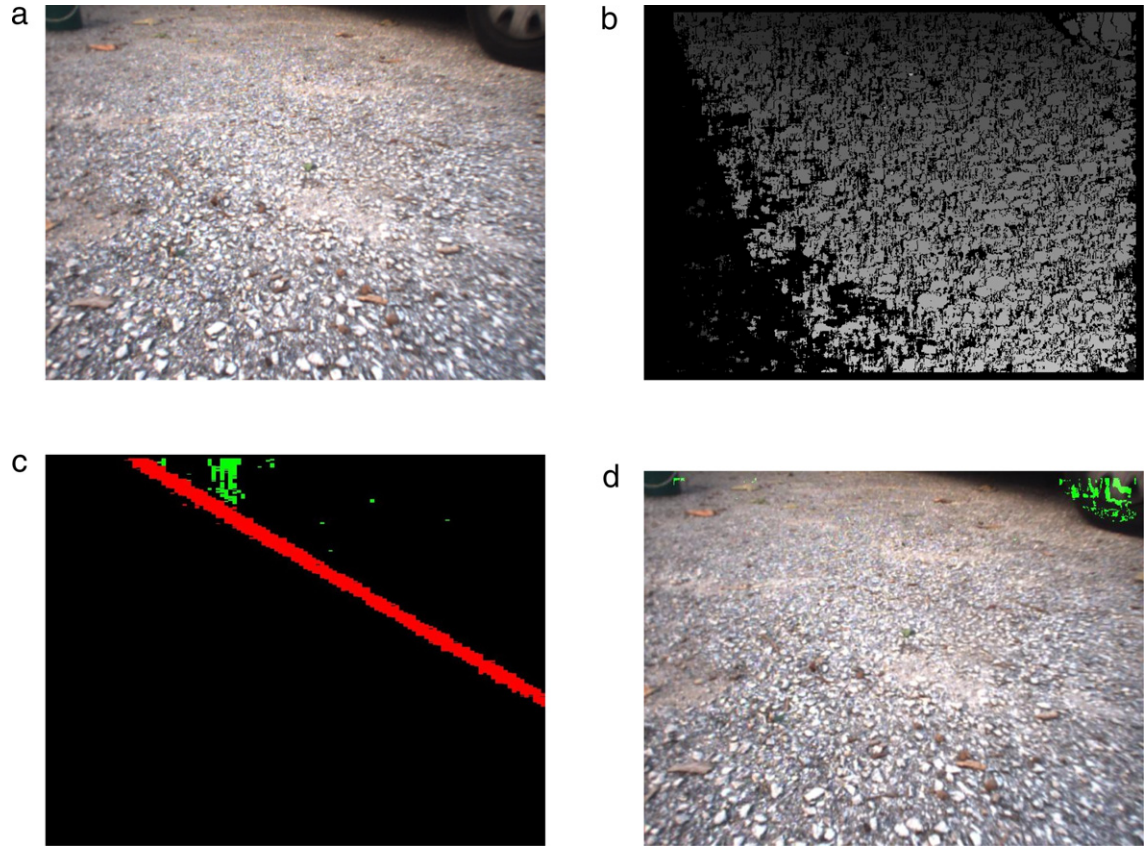


Fig. 3. (a) A left-reference image during the robot's exploration, (b) the respective disparity image, (c) the v-disparity image where the ground plane is modeled with red color, while the obstacles above the ground are indicated by green color and (d) the highlighted obstacles on the left-reference image. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

by modeling the terrain in front of the robot. More precisely by using the disparity map calculated on each step, a reliable v-disparity [37] image can be computed as shown in Fig. 3(c). The v-disparity image, being a 2D statistics of the disparity map allows direct distinguishing between the ground plane and the obstacles. In particular, the ground plane in the v-disparity image is modeled by a linear equation. A tolerance region extending on both sides of the linear segment of the terrain is defined and any point outside can be safely considered as belonging to an obstacle. The linear segments denoting the terrain and the obstacles are depicted on the v-disparity image of Fig. 3(c).

The global map update procedure considers: (i) detecting the new objects that have been added to the scene since its last update with a full-circle point cloud and (ii) filling the map with regions that have not been registered during the original map acquisition, due to occlusions by huge obstacles such as pillars, and cars. The detection of the obstacles is performed in every motion estimation. More precisely, for any new stereo pair, the v-disparity image is computed and the detected obstacles are utilized to update the global map. The result is a sparse point cloud which contains the observed obstacles distinguished from the ground plane. In the next step, these point clouds are further transformed considering the robot's pose – incrementally calculated during the motion estimation – and placed into the global map.

2.3.2. SVM local path planning — SVMLP

The intuition of the proposed local planner stems from a trivial geometrical remark, i.e. the safest trajectory for a robot cruising among obstacles is the one that preserves the maximum margin from all of them. Considering the case where the current position of a robot and its desired one are known, then, the trajectory should also be constrained accordingly. Moreover, the obstacles – given as distinctive points – need to be clustered in a way that the minimum distance between those groups is greater that the robot's embodiment. The *geometrical interpretation of the SVM* keeps up with the aforementioned intuition, as explained in the following illustrated example. Let us assume that Fig. 4(a) depicts a set of points clustered in two different groups, owing to their in-between Euclidean distance being greater than the robot's embodiment along with their corresponding convex hulls. The solution to the SVM's dual problem for the respective task derives from the hyperplane that bisects the linear segment joining the two nearest points between the convex hulls of the clusters. That is, the acquisition of the maximum margin hyperplane is equivalent to the pursuit of the two nearest points between the respective convex hulls, as illustrated in Fig. 4(b). The formation of convex hulls is a common property between the SVMLP and Voronoi-based methods. However, several drawbacks existing in Voronoi based methods do not appear in the proposed one. As an example, the work in [16] mentions the so-called "weak meet points" where small errors in sensor reading may lead to drastic changes in the relationship among all meet points. The latter implies significant alternations in the robot's world perception. An improved version was proposed in [17], yet this work strictly depends on accurate place categorization and, moreover, the authors consider indoor cases only. Regarding the comparison between Voronoi based methods and the proposed approach, in the first ones a robot may navigate between the junctions upon the branches that connect them, which is of course a safe path. However, in our approach, the hybrid combination of a GPP and the SVMLP provides both the shortest path towards a goal position – as a property of the GPP – and a continuous smooth trajectory among the points of

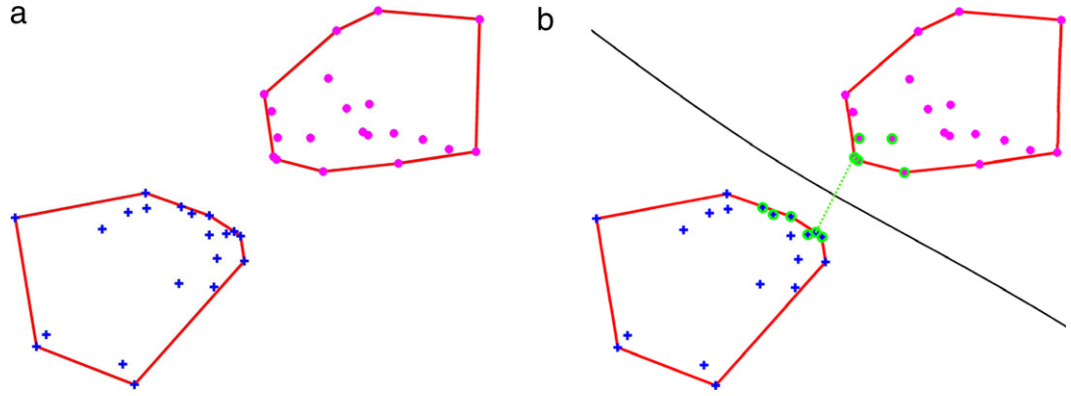


Fig. 4. (a) A set of points clustered in two groups and their corresponding convex hulls. (b) The extracted hyperplane bisecting the line segment that unites the two closest point between the convex hulls. The red line indicates the respective convex hulls of the two clusters, the green points are the support vectors, the dashed green line segment conjuncts the two nearest points, whilst the black one is the separation hyperplane. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the path, which is the safest one. Additionally, the fact that the SVM's cost function is a strict convex one and the respective inequality constraints comprise linear functions, pledges that any local minimum is also a global one and sole. In terms of local path planning, the resulted SVM solution which in this work is considered to be the robot's trajectory is optimal and unique. It is worth noting that the GPP is mathematically proved to produce the optimal path within the scene, the SVMLP is responsible for providing the optimal trajectory in terms of clearance which it might be suboptimal in terms of the traveled distance among the consecutive points of the GPP.

Accordingly, SVMLP operates on the updated 2D map $\mathcal{M}_2$ and, for the derivation of the continuous trajectory, it considers the points within a rectangle originating by two points, viz. the current position and the successive point of the GPP. Let $\mathbf{p}^i, \mathbf{p}^{i+1} \in \mathbb{R}^2$ be those two points, then the rectangle's orientation follows the direction that those two points define. Moreover, the two sides of the rectangle parallel to the line defined by $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$, have length T_{dp} equal to $2 \times \|\mathbf{p}^i - \mathbf{p}^{i+1}\|$. The length of the other two sides is determined by a parameter T_{nl}, whilst $\mathbf{p}^i, \mathbf{p}^{i+1}$ constitute the midpoints of those sides respectively. From the rectangle dropped, the four points $\mathbf{r}_a, \mathbf{r}_b, \mathbf{r}_c, \mathbf{r}_d$ are the intersection of the sides and result in a straightforward fashion. These four points are used to estimate the subset of points in the $\mathcal{M}_2$ map that lie within the rectangle. For every point $\mathbf{m}_j \in \mathcal{M}_2, j = 1, 2, \ldots, N_P$, where N_P is the number of points of the $\mathcal{M}_2$, the $\mathbf{m}_j$ lies within the rectangle if and only if

$$0 \leq \overrightarrow{\mathbf{r}_a\mathbf{m}_j} \cdot \overrightarrow{\mathbf{r}_a\mathbf{r}_b} \leq \overrightarrow{\mathbf{r}_a\mathbf{r}_b} \cdot \overrightarrow{\mathbf{r}_a\mathbf{r}_b}$$
$$\text{and} \qquad (1)$$
$$0 \leq \overrightarrow{\mathbf{r}_a\mathbf{m}_j} \cdot \overrightarrow{\mathbf{r}_a\mathbf{r}_d} \leq \overrightarrow{\mathbf{r}_a\mathbf{r}_d} \cdot \overrightarrow{\mathbf{r}_a\mathbf{r}_d}.$$

The geometrical points inside the rectangle form a list $\mathcal{L} \subseteq \mathcal{M}_2$ which is utilized for the calculation of the trajectory. The points $\mathbf{p}^i, \mathbf{p}^{i+1}$, along with the defined rectangle and the $\mathcal{L}$ set of points are annotated on a map segment constructed by real data as illustrated in Figs. 5(a) and 6(a).

The case where the list is empty, i.e. $\mathcal{L} = \varnothing$, implies that the area of interest is an obstacle free one and the robot may travel from point $\mathbf{p}^i$ to $\mathbf{p}^{i+1}$ utilizing the corresponding line segment they define, since it constitutes the shortest route. However, in most cases $\mathcal{L}$ is a non-empty set, thus a non-linear trajectory needs to be determined. In this case, the proposed method determines among which obstacles the robot is able to traverse, considering its embodiment. For this goal to be achieved, the methodology groups the points $\mathbf{o}_i \in \mathcal{L}$ such as the closest distance between the clusters is greater than a threshold T_{re}. This threshold should consider the dimensions of the robot along with a tolerance that assures the avoidance of collision. Moreover, threshold T_{re} is a tractable one, rendering its regulation a straightforward task. The clustering scheme utilized for the grouping of $\mathbf{o}_i$ is an agglomerative hierarchical one. The metric used for the similarity matrix is the Euclidean distance, i.e. $d = \|\mathbf{o}_i - \mathbf{o}_j\|$, where $\mathbf{o}_i, \mathbf{o}_j \in \mathcal{L}, i \neq j$ since the points correspond to real world coordinates. The linkage criterion that determines which clusters are merged is the single-linkage one:

$$C_k^{new} = C_k \cup C_l \quad \text{if } \min\{d(\mathbf{o}_i, \mathbf{o}_j) \leq T_{re} : \mathbf{o}_i \in C_k, \mathbf{o}_j \in C_l\} \qquad (2)$$

Eq. (2) declares that any two clusters in a certain iteration of the hierarchical agglomerative scheme can be merged if the closest distance between the points of clusters C_l, C_k is less than T_{re}. The extraction of such clusters is depicted in Figs. 5(b) and 6(b). The derived clusters are labeled by their position with respect to the conceptual line, which is defined by the current position and the desired one. The points $\mathbf{p}^i, \mathbf{p}^{i+1}$ are used in the formation of the respective linear function $g(\mathbf{o}) = \mathbf{v}^T\mathbf{o} + v_0$, where $\mathbf{v} \in \mathbb{R}^2$ is the orthogonal vector to the line and $v_0 \in \mathbb{R}$ the bias term. For every cluster C_k if all its points $\mathbf{o}_i \in C_k$ are placed in the same half-plane, then the cluster C_k is labeled with the corresponding sign. Regarding the case where a cluster C_k attains points that lie on both semi-planes of $g(\cdot)$, i.e. $\exists\, \mathbf{o}_i, \mathbf{o}_j \in C_k, i \neq j : g(\mathbf{o}_i) \cdot g(\mathbf{o}_j) \leq 0$, then the labeling of C_k is computed by the respective sign of its mean value $\boldsymbol{\mu}_{C_i} = \frac{\sum_i \mathbf{o}_i}{|C_k|}$. Of course, if all points of a cluster lie in the same half-plane, so does the mean value. As mentioned above, this scheme occurs for all clusters, deducing to the attribution of labels for all points in $\mathcal{L}$. Consequently, the labeling of C_k is computed as follows:

$$Label(C_k) = sgn(g(\boldsymbol{\mu}_{C_i})). \qquad (3)$$

The following step addresses the coercion of the continuous trajectory through points $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$. This is accomplished by imposing virtual points parallel to $g(\cdot)$ on both its sides. The virtual points are appended nearby $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$, compensating thus the robot's embodiment, while their number on each side is N_v and their labels come of in a straightforward manner. The labeling of the set $\mathcal{L}$ and the addition of the virtual points are illustrated in Figs. 5(c) and 6(c). The labeling of the points leads to a non-linear separable problem. Consequently, the method uses the hard maximum margin scheme of SVM, along with the second order polynomial kernel. This setup deduces the following Lagrangian primal form of the quadratic constrained problem:

$$L(\mathbf{w}, w_0, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N_v} \lambda_i[y_i(\mathbf{w}^T\phi(\mathbf{o}_i) + w_0) - 1] \qquad (4)$$

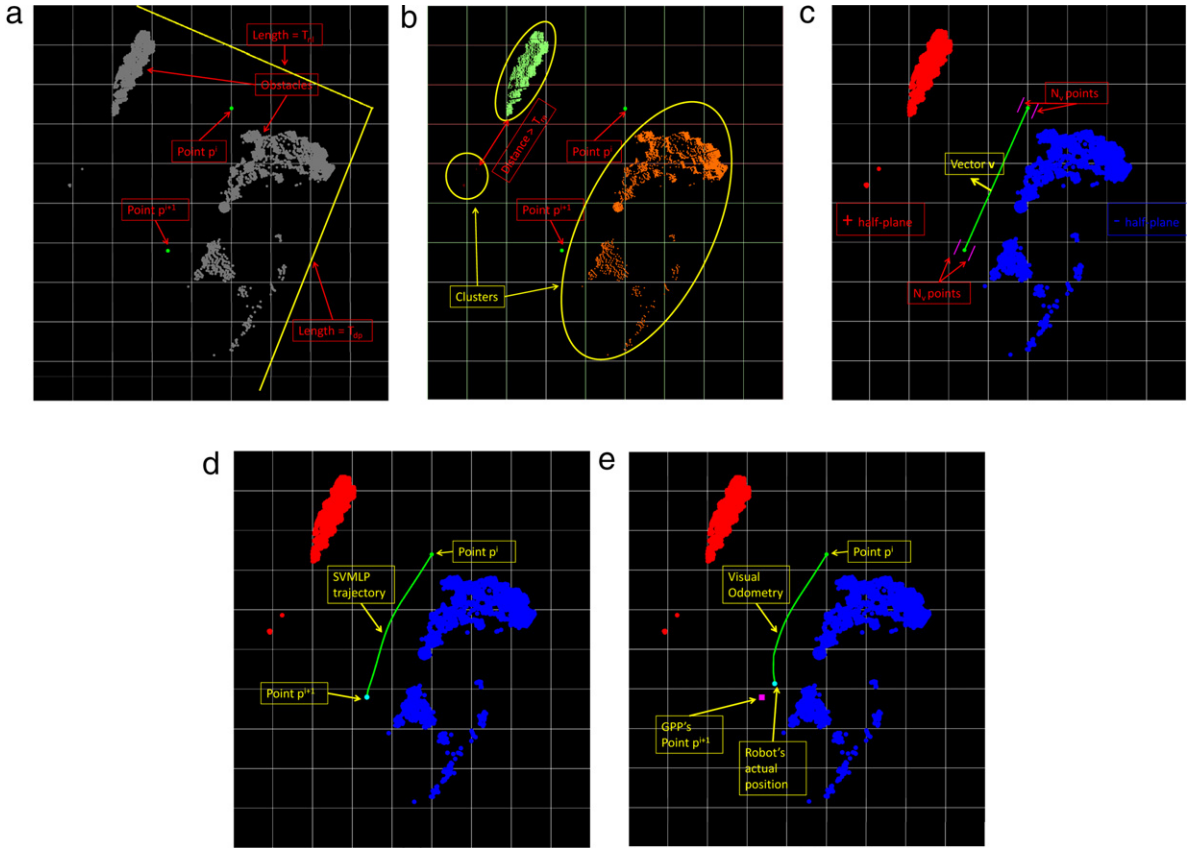


Fig. 5. (a) The points $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ are needed to compute one side of the virtual rectangle T_{dp}, while the other side's length equals T_{nl}, a user defined threshold. The depicted points constitute the $\mathcal{L}$ set of obstacles. (b) The clustering result of the $\mathcal{L}$ set is based on the T_{re} parameter. The minimum distance among the clusters is sufficient for the robot's navigation. (c) The virtual points, which are parallel to the $g(\cdot)$ line, are shared equally on both half-planes with respect to the $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ points. At this stage, the labeling of the clusters occurs relying on the $g(\cdot)$ function. (d) The continuous trajectory is computed by the SVMLP component. The extracted course is a smooth one due to the selected kernel. Moreover, because of the maximum margin property it "leans" towards the middle of the classes, whilst at the same time obeys the restriction to pass through $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ points. (e) Execution of the SVMLP path by the robot, illustrating its respective displacement due to drifting on the robot's stepper motors. Note that the green line indicates the output of the motion estimation (localization based on the VO component) and the purple square indicates the target point as dictated by the GPP.

which provides the following Wolfe dual representation:

$$\max_{\lambda} \quad \left(\sum_{i}^{N_v} \lambda_i - \frac{1}{2}\sum_{i,j}^{N_v} \lambda_i \lambda_j y_i y_j K(\mathbf{o}_i, \mathbf{o}_j)\right) \tag{5}$$

$$s.t. \quad \sum_{i=1}^{N_v} \lambda_i y_i = 0$$

where $\mathbf{w}$ is the vector containing the coefficients of the function and w_0 the corresponding bias term, λ_i are the Lagrange multipliers, y_i the labels of the $\mathbf{o}_i$ points. Additionally, $\phi(\cdot)$ is the mapping function to the second order polynomial space and $K(\cdot, \cdot)$ the respective kernel. The derived non-linear hyperplane constitutes the safe continuous trajectory of the robot from its current location to the point dictated by the GPP. An example of such trajectories is depicted in Figs. 5(d) and 6(d). Note that Figs. 5(e) and 6(e) illustrate the robot's trajectory as it is estimated by the VO component. The fact that the robot does not reach its target location but ends its route earlier can be explained if we consider the drift of the stepper motors attached on the robot's wheels. How close the robot will reach its target location depends both on the accuracy and sensitivity of the stepper motors and of course on the quality of the terrain. In a slippery terrain the robot will loose many tics on its motors, however, on a terrain with satisfactory grip the robot will loose less tics. That is also the reason that indicates the necessity of the VO within our framework.

2.4. System integration

This subsection describes how all the presented components intertwine with each other. In general, it presents the aftereffect of the map update regarding the re-planning and the repetitious screening of the robot's localization in comparison with the dictated trajectory, as depicted in Fig. 7. The proposed system initiates with the acquisition of a 360° point cloud followed by the steps that produce the $\mathcal{M}_2$ map. Based on this map, the robot initiates its navigation towards the goal point by applying the D* Lite while the sequence of points determined by the GPP are used by the SVMLP to extract the continuous trajectory. The GPP is triggered to re-plan by considering the current localization of the robot whenever the $\mathcal{M}_2$ map is updated. There are two cases, by which the $\mathcal{M}_2$ map is revised; the first one involves the acquisition of another 360° point cloud, which is followed by the registration procedure as described in Section 2.1, while the second one considers the obstacle detection executed in every motion estimation step, as shown in Section 2.3.1. Besides the map revision, the robot pose update may result in re-planning. These effects may occur due to two reasons: the first one involves the case where the robot is pointed out of the rectangle defined in Section 2.3.2 and the second one compares the coordinates of the point $\mathbf{p}^{i+1}$ against the localization of the robot after concluding the trajectory imposed by the SVMLP. In the event that the actual position of the robot is significantly different than the imposed one (i.e. $\mathbf{p}^{i+1}$), the D* Lite re-plans according to its

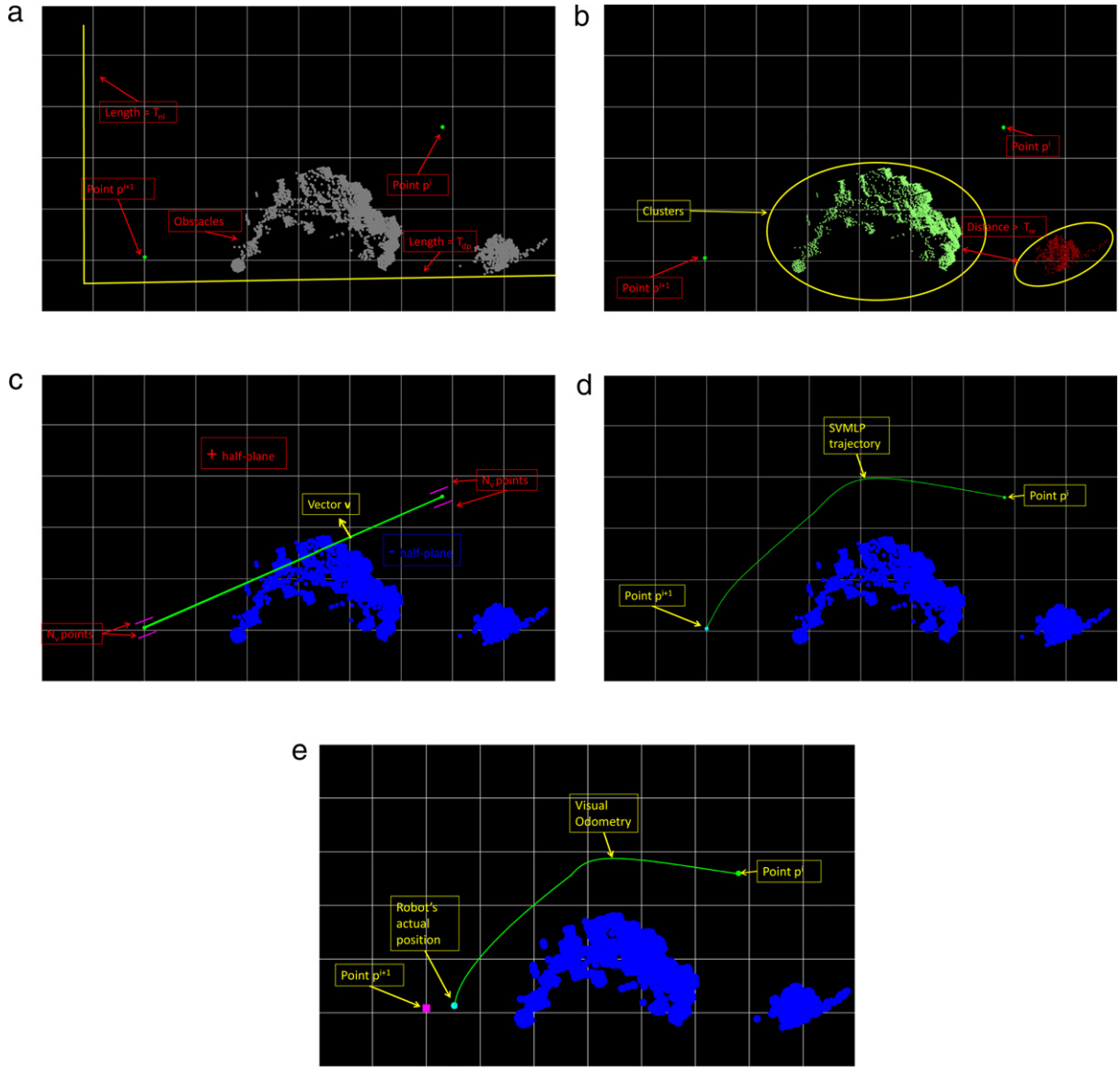


Fig. 6. (a) The points $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ are needed in order to compute one side of the virtual rectangle T_{dp}, while the other side's length equals T_{nl}, a user defined threshold. The depicted points constitute the $\mathcal{L}$ set of obstacles. (b) The clustering result of the $\mathcal{L}$ set is based on the T_{re} parameter. The minimum distance among the clusters is sufficient for the robot's navigation. (c) The virtual points, which are parallel to the $g(\cdot)$ line, are shared equally on both half-planes with respect to the $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ points. At this stage, the labeling of the clusters occurs relying on the $g(\cdot)$ function. (d) The continuous trajectory is computed by the SVMLP component. The extracted course is a smooth one due to the selected kernel. Moreover, because of the Maximum Margin property it "leans" towards the middle of the classes, whilst at the same time obeys the restriction to pass through $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ points. (e) Execution of the SVMLP path by the robot, illustrating its respective displacement due to drifting on the robot's stepper motors. Note that the green line indicates the output of the motion estimation (localization based on the VO component) and the purple square indicates the target point as dictated by the GPP.

current position, as computed by the VO. Regarding the SVMLP, the $\mathbf{p}^i$ is always the one computed by the VO, while the $\mathbf{p}^{i+1}$ is the one administered by the GPP.

At this point, it is worth discussing the resolution of the grid concerning the GPP. It has already been mentioned in Section 2.2 that GPPs operate in a discrete level thus, in this paper, the distance of the centers between two successive cells in the occupancy grids correspond to 0.2 m in the real world. The GPP does not use the resolution of the derived $\mathcal{M}_2$ in full extend, but rather provides a rough trajectory in terms of points. The reason behind this choice is twofold: firstly, the computational burden of the GPP is significantly reduced in comparison with the usage of a much denser 0.01 m-resolution grid. Secondly, the dimensions of the grid are utilized to check whether the position of the robot is close enough to the dictated end position $\mathbf{p}^{i+1}$. In the case where the resolution of the GPP is in the order of magnitude of cm, then the robot would probably fail to reach $\mathbf{p}^{i+1}$ with an accuracy being in the same order, which in turn would lead the GPP to re-plan almost every time. It is also important to mention that currently we rely on the grid resolution to evaluate whether the robot has reached its target location. However, in our next step we plan to measure the Euclidean distance among the target point and the current position of the robot which is calculated by the VO, by applying a large enough metric that considers a circular goal region. Moreover, the SVMLP uses the detailed resolution of the map to compute the continuous trajectory. Although the GPP attains a 0.2 m resolution, the SVMLP would not follow iteratively all the derived trajectory points of the D* Lite. On the contrary, at a position $\mathbf{p}^i$ the SVMLP seeks the next point $\mathbf{p}^{i+1}$ on the GPPs' route that is about 1 m away. Fig. 8 illustrates the proposed integrated system operating in an outdoors scenario.

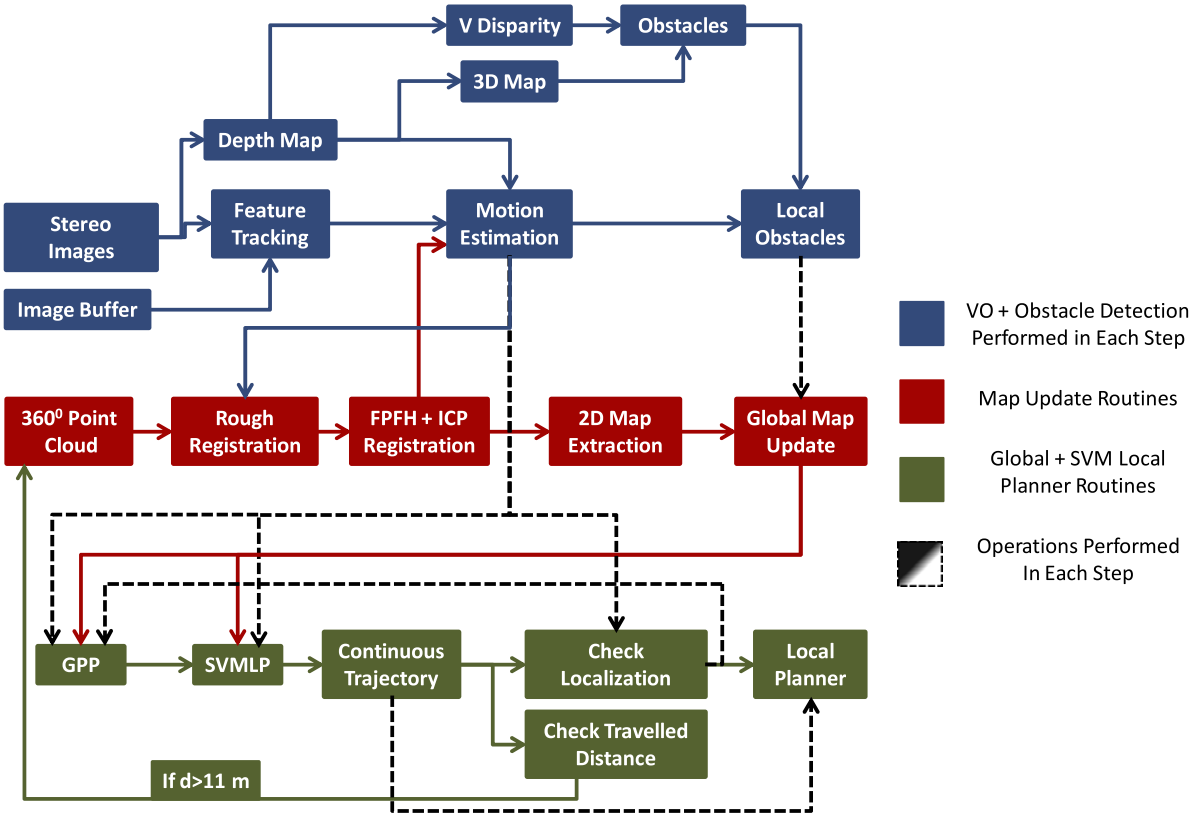


Fig. 7. The architecture of the autonomous navigation system.

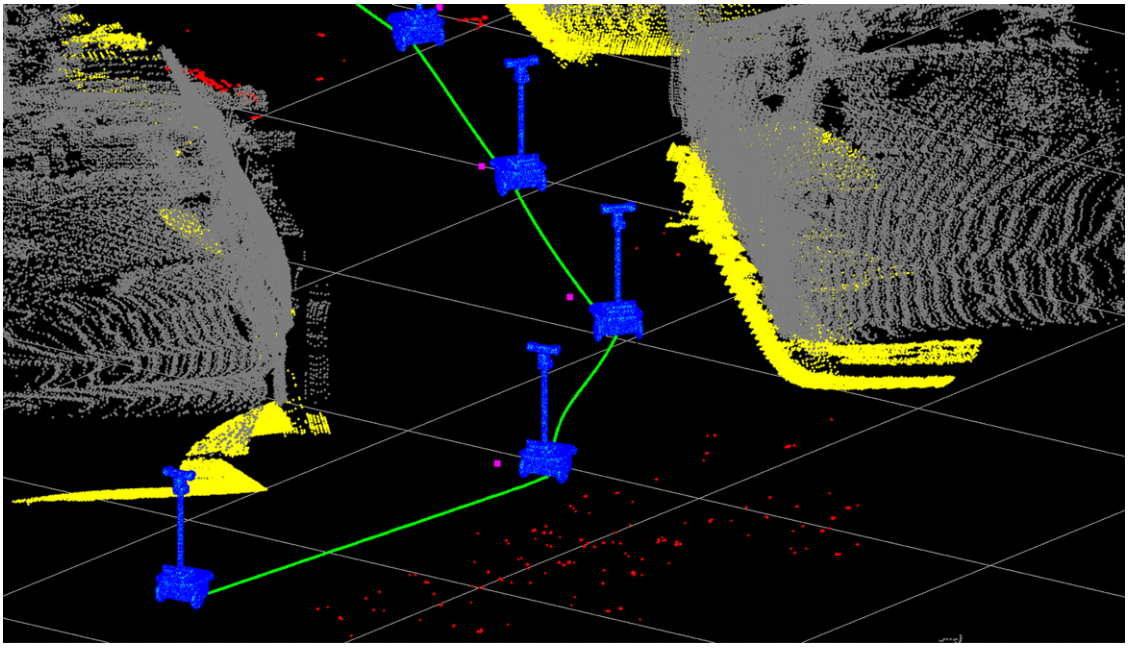


Fig. 8. Figure illustrating the robot's operation using the proposed architecture. The yellow points correspond to the 360° point cloud after the respective processing, the red ones to the obstacles detected in each step along the course of the robot and the gray are yellow ones before the top-down projection. The latter is used strictly for visualization purposes. The purple squares designate the $\mathbf{p}^{i+1}$ points, imposed by the GPP while the schematic robot in the figure illustrates the actual position of the robot. The green curve is the robot's continuous trajectory. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3. Experimental results

This section presents the evaluation of the proposed autonomous navigation system utilizing a custom robotic setup. The experiments were conducted outdoors and proved the suggested method exhibiting remarkable behavior both in terms of safe navigation and detailed map formation. The system handled successfully the cruising in semi-structured spaces, providing feasible and smooth trajectories, which were equidistant from the lateral the obstacles. At this point it should be mentioned that, even though similar methods utilizing SVM for local planning have already been developed in the past, a direct comparison with any

such method is not feasible. The existing methodologies operate in an open loop fashion, inferring solely for the currently observed scene. However, the proposed technique performs on long range data, in a closed loop fashion, as part of an integrated robotic system. Here, multiple inferences are made in each step by combining different sensory inputs leading the robotic platform to safely reach its target in a long range route.

3.1. Robot setup

The utilized robot is a custom made one built upon ERA-Mobi platform (Erratic) produced by Videre Design. This is a differential wheeled robot, with dimensions of 40 × 41 × 15 cm. Its maximum speed is limited to 2 m/s and the respective maximum payload is 20 kg. The platform is equipped with a low-end on-board PC bearing an Intel® Core™ 2 Duo processor at 2.0 GHz with 4 GB of RAM. Upon this platform a metallic skeleton has been constructed to mount the utilized sensors which are put to use for the mapping and the localization of the developed algorithm. Regarding the mapping component the SICK® LMS500 PRO 2D laser scanner was used, which is capable of providing range readings on a plane with high level of accuracy. In more detail, the field of view is 190°, the angular resolution varies within the interval of [0.1667°–1°], the scanning frequency alters in [25–100] Hz interval while the scanning range is up to 80 m. Owing to this being a 2D laser scanner, it is attached on the pan–tilt unit (PTU) PW-70 by SCHUNK®, which enables the laser scanner to rotate in order to achieve vertical scanning, thus perceiving 3D information of the environment. The angular resolution of the scanner is set to 0.1667° and the 3D scan is produced by performing a 360° sweep. For the localization component the Point Grey® Bumblebee2™camera has been utilized, which is a compact construction and retains precise alignment – with a fixed baseline of 12 cm – between the two vision sensors. Thus, it allows exact 3D features extraction to be used for the robot's visual odometry module [36]. Considering the physical arrangement of the proposed setup, the complex comprising the laser scanner and the PTU is placed 1.12 m above the ground ensuring that the 3D scan will acquire observations higher than the robot's embodiment, thus facilitating its navigation affordances in the perceived environment. The localization stereo camera is placed 30 cm above the ground, so as to ensure accurate 3D features tracking. Fig. 9 depicts the robot during its deployment in an outdoors scenario. Since the different devices do not communicate with each other and an external triggering mechanism is not available to synchronize them, ROS (Robot Operating System) was chosen to fulfill this task, by means of timestamps to the messages published from each device. An additional advantage of ROS is that it supports drivers for all the aforementioned devices, including the robot platform. In particular, the laser scan assembling procedure comprises the rotation of the scanner and its elevation as well, while the stereo pairs acquisition is performed on demand. As a result, the derived 3D point cloud embodies the respective transformations, facilitating this way its manipulation and further processing.

Regarding the acquisition of the full-circle point cloud, during the 360° panning movement the system captures the respective laser scans with an angular resolution of 1°, i.e. each 360° sweep consists of 360 individual 2D scans. In order to acquire a dense point cloud, where the points lay at most 0.2 m away from each other, the 360° scan procedure is repeated approximately every 11 m. The computation of this distance is derived through simple geometrical calculations. *In order to ensure that the proposed framework will operate both on flat and rough terrain, two important facts have been considered. The first one is that the global map is updated with a new 360° scan when the robot has traveled 11 m, which is a decent compromise among the resolution of the resulting 3D map and the scanning time needed. In this way any abnormalities of the ground are modeled in the 3D map early and, thus can be avoided during the execution of the GPP. However, the small obstacles that filtered out during the truncation of the 3D map (for the formation of the 2D grid) are reconsidered with great resolution in the v-disparity procedure and treated as obstacles by the SVMLP. The execution of the local obstacle detection routine is performed with 1 Hz ensuring thus maneuverability in uneven terrain. The second fact that allows the proposed algorithm to operate on rough environments with great alterations in the robot's pose is the design of the VO algorithm itself. The utilized localization routine exhibits maximum performance for small angular and translational changes, which is executed in 1 Hz with 3–6 cm sampling. In cases that two successive instances witness robot's rotation greater than* $[-10, 10]^\circ$*, then an additional refinement step is performed by executing the ICP among the point clouds of the currently detected obstacles and the existing global 3D map. Thus, when the robot travels in uneven terrain the subsequent changes in the 3D map are taken into consideration.*

Fig. 9. The robotic platform ERA-Mobi equipped with a LMS500 PRO laser sensor and a PW-70 pan–tilt unit and a Bumblebee2™stereoscopic camera.

3.2. Parameters selection

The system's parameters concern the point cloud registration, the SVMLP and the localization ones. Regarding the point cloud registration parameters, they are expressed in terms of meters and due to the initial rough alignment provided by the VO component, their values were comparatively small and remained static for all the evaluated scenarios. In more detail, the registration procedure estimates the key-points on both point clouds and for that computation a uniform grid of 1 m length is applied. Then, for every point the respective normal vector is computed, relating on the points within a neighborhood of a 0.5 m radius. Both the key-points and the normals are essential for the calculation of the FPFH features, which is followed by the location of FPFH correspondences between the point clouds. The last parameter is the rejection of correspondences having distance greater than 1 m.

Table 1
Average duration of point cloud registration from various trials. The first column provides the execution time without the rough alignment step while the second one the duration when using the localization.

	Without rough estimation ± std (in seconds)	With rough estimation ± std (in seconds)
FPFH	86 ± 18.24	12 ± 0.03
ICP	53 ± 9.31	9 ± 0.01

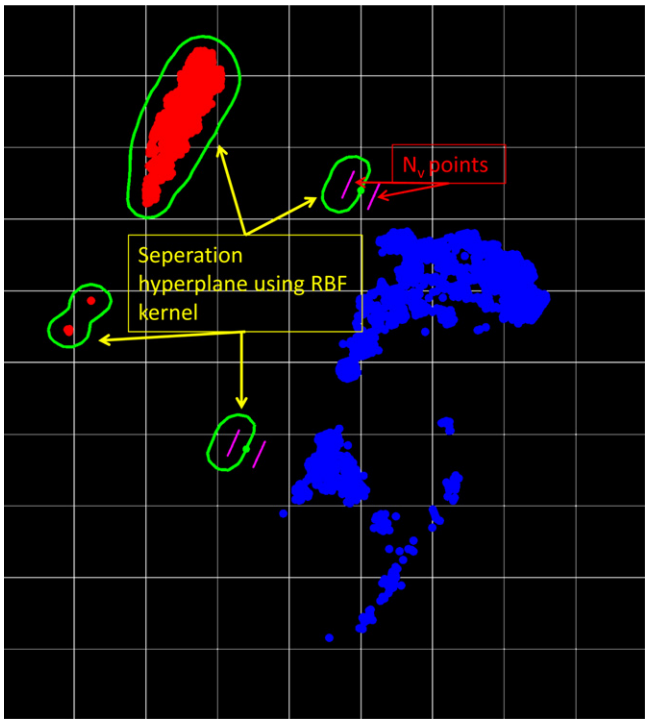


Fig. 10. The separation plane in the SVMLP component when instead of the 2nd order polynomial kernel, the RBF one is used, which is in accordance with the example of Fig. 5.

The point cloud processing also relies upon several parameters, however their regularization is straightforward since it depends on static sizes, namely the bounding box of the robot. The only parameter that may not come in such an obvious fashion is the one posed in the voxel grid filtering. The latter, is essential because it removes duplicated points, resulting in a more efficient system, whilst the risk of extracting maleficent trajectories is compensated when the edge of the grid attains significantly decreased value.

Concerning the SVMLP component, T_{nl} defines the rectangle in which the SVMLP takes place. The respective parameter is adjusted based primarily upon the dimension of the robotic platform. Consequently, this parameter holds the balance between the space in which the SVMLP operates and the computational burden of SVMLP, since in most cases holds that the wider the area of interest the more points need to be considered. From several experimental cases we concluded that if T_{nl} acquires values over triple its diagonal, then SVMLP operates without re-planning. Another essential threshold of the respective module is the T_{re}, which is involved in the clustering of the $\mathcal{L}$ set. Substantially, it ensures whether the robot can traverse between two obstacles and theoretically any value greater than its diagonal is sufficient for this task. However, due to small errors that occur either due to hardware restriction, drifting while the robot moves etc., we set its value equal to 150% the robot's diagonal length. As far as the SVM parameters are concerned, the utilization of a hard margin approach not only avoids the determination of the relaxation parameter but also attains a physical meaning. The insight behind this idea is that the separation plane cannot allow "misclassification" of points since it would lead the robot to collide with them. Moreover, due to the fact that the labeling of the points is in proportion with the derived clusters, the separability of them (points) is secured. Another issue that arises is the selection of the kernel. There are several kernel choices, namely the radial basis function (RBF), the Hyperbolic Tangent and the Polynomials. Yet, the application of either the RBF or the Hyperbolic Tangent kernel might lead to an improper nonlinear separation plane on the original (2D) input space, in terms of robot's trajectory. Both the aforementioned options tend to overtrain – an effect that is more frequent in the particular problem due to the structure of the data – and derive discontinued planes in the input space. The behavior of those kernels strictly relies on their respective parameters, especially in the case of the RBF one, which is remarkably sensitive regarding the γ values, i.e. the grade of influence of a single training example. An example case where the γ parameter leads to over-train is illustrated in Fig. 10. However, the 2nd degree polynomial kernel is a low degree mapping closer to the input space, less susceptible to over-train in comparison with other kernels, while it generates trajectories feasible for the robot, i.e. ones that are not excessively curved or unnatural. In fact, third and fourth degree kernels were also employed and solved competently the local planning problem. All the aforementioned kernels derived natural trajectories, and the differences lie in the curvatures of the trajectories. The 2nd degree polynomial kernel was preferred due to the fact that it demonstrates less curvature. Last the parameter selection of the localization component has been thoroughly discussed in [35].

3.3. Algorithm assessment

The respective components are individually assessed and the full appraisal of the system follows. The GPPs and the LPPs rely on the robot's position and the description of the surrounded environment to deduce a collision free path. Moreover, in this paper, both the localization component and the point cloud registration one are interlinked, correcting one another. That is, the point cloud registration relies on the localization counting on the VO component, which comprises the rough alignment between the point clouds. Afterward, when the refinement step of the registration component concludes, the corresponding transformation updates the robot's localization too. Therefore, the refinement step, consisting of the FPFH features and the ICP, was also evaluated individually to determine whether they are sufficient to provide accurate registration. This experiment exhibits two major drawbacks, the first one being the application of significantly greater values in the respective parameters, described in Section 3.2. The aftereffect was the considerable amount of time needed for the two procedures to converge, while there were cases that both the FPFH and the ICP failed to conclude due to error minimization rather due to over-exceeding the number of steps, which is simply a "safety valve" that terminates the respective procedure. In those cases the registered point cloud was erroneous. The rough alignment step (VO engagement) alleviated those issues by substantially reducing the time needed for the entire registration procedure and delivering correctly registered point clouds. The time duration for the registration is summarized in Table 1. The second drawback is that even in the case that convergence is accomplished, the results will be inferior comparing to the ones involving a rough alignment. In Fig. 1 the reduction of error in registration with a refinement step

Table 2
Qualitative comparison of SVM-based path planning methodologies.

	Miura [26]	Tennety et al. [28]	Qingyang et al. [29]	Proposed work
Complexity	$\mathcal{O}(C_{T1} \cdot n^3)$	$\mathcal{O}(n^3 + 2n)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
Kernel Type	RBF	RBF	RBF	2nd order P.
Closed Loop	–	–	–	✓
GPP	–	–	–	✓
Robot Embodiment	–	–	✓	✓
Non-Convex Obstacles	✓	–	✓	✓
Evaluation data	Simulated	Simulated	Real	Real
Resolution	Not available	Not available	Not available	0.2 m
Ground truth	Not available	Not available	Not available	Not available
Distance traveled	–	–	Not available	113.24 m

is depicted, whilst Fig. 12 illustrates the $\mathcal{M}_2$ map corresponding to Fig. 11(c). The $\mathcal{M}_2$ map illustrated in Fig. 12 comprises solely the obstacles derived by the 360° scans, however, during the robot's travel to a goal position the map is also appended with obstacles exposed by the VO component.

By qualitative comparing the proposed local planner with the one in [26], the latter is a theoretical approach, operating in an open loop fashion, where the robot's embodiment is not taken into consideration. Moreover, the work in [26] proposes a soft margin approach combined with the RBF kernel, raising the issues of colliding with obstacles and creating discontinuous paths. The computational complexity is of $\mathcal{O}(C_{T1} \cdot n^3)$, where $\mathcal{O}(n^3)$ is the complexity of the SVM and C_{T1} is the number of iterations for alternative path generation. This is significantly greater than the one in our proposed work, where the SVM formulation is solved only once. The work in [28] was also evaluated in a simulation environment following an open loop strategy. The complexity in addition to the one introduced by the SVM is burdened with the respective one of the nearest neighbor, i.e. $\mathcal{O}(2n)$. Moreover, the connection of this work along with a GPP or the embodiment of the robot is absent. Last, the planner introduced in [29], leaves the issue of the γ parameter of the RBF kernel unresolved, while the path subdivision scheme employed within the path planning is unable to deal with non-convex obstacles. Table 2 provides a comparison of all the related algorithms and the proposed one while Table 3 a quantitative presentation the computational cost, the area they cover and the frequency of occurrence of the subroutines involved within the proposed work. In Miura's paper, the respective times are given regarding the path generation as well as the total planning time. More precisely, the average time for the path generation is about 3.3 s while the mean total planning time is approximately 80 s. It is worth noting that since the corresponding experimental results took place in form of simulation computation times about the formation of the map and the respective obstacles were not provided. Moreover, in those planning problems there were only 10 obstacles, a scenario which is not encountered that often in real applications. In the proposed scheme in order to reduce the computational cost only a neighbor around the successive points $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ is considered, while its corresponding area is regularized via the T_{nl} parameter. As a result, the cost of the frequently executed tasks, such as the trajectory calculation is decreased. The most time expensive subroutine is the registration one and occurs every 11 m in order to maintain a resolution of 0.2 m, which is also the resolution of the GPP. Of course, a less detailed map could be made, reducing the frequency of occurrence as well as the analysis on which GPP is performed. Depending on the environment and the arrangement of obstacles such a map may lead to feasible but less optimal solutions.

Besides the individual valorization of the components, the overall system was evaluated on long range outdoor scenarios as well. Several different environments have been utilized to complete the validation procedure, ranging from simple wide open areas consisting of a few trees and rocks to particularly challenging ones,

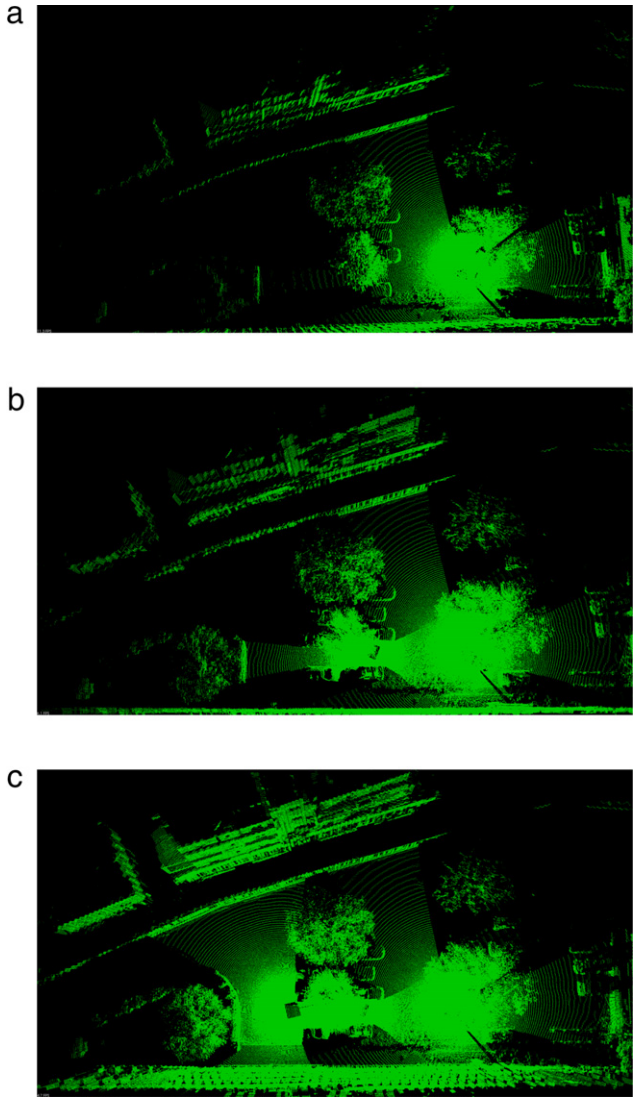


Fig. 11. (a) A 360° scan (b) the point cloud of 13(a) registered and merged with another 360° scan (c) the point cloud of 13(b) registered and merged with another 360° scan.

incorporating trees, rocks, buildings, railings, pavement and cars. The results presented here correspond to the latter case, while the robot started up being surrounded by cars and walls. The starting position was selected in purpose with the aim to disturb the line of sight of the robot. As a result, the shortest route to the goal position was between cars. Having insufficient visual information, the robot initially was designed to cross between a pair of cars, where their intermediate space was insufficient for the robot to traverse. Towards its journey the map was accordingly enriched

Table 3
Quantitative presentation of the computational cost, the area they cover and the frequency of occurrence of the subroutines involved within the proposed work.

Subroutine	Computation time (s)	Coverage	Frequency of occurrence
V. O.	≤ 1	2.5 m	Every 1 s
Registration	38.7 ± 5.8	80 m^2	Every 11 m
SVMLP	1.65 ± 0.2	Rectangular w.r.t. T_{nl}	Every 1 m
D* Lite	2.5 ± 0.3	Entire route	Every 11 m or when re-planning

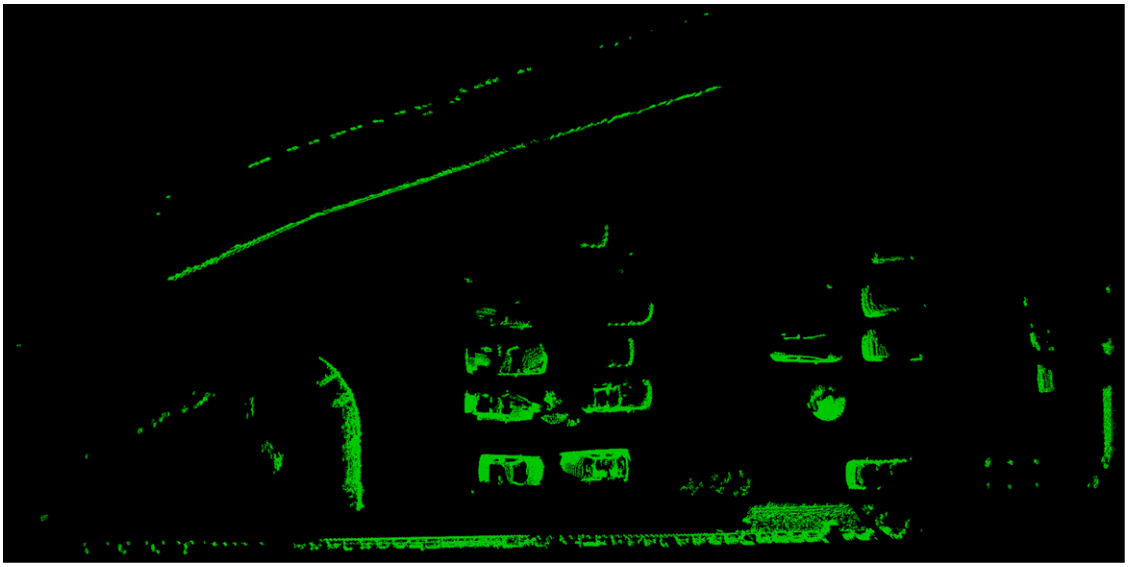


Fig. 12. The respective $\mathcal{M}_2$ map of Fig. 11(c) consisting of obstacles revealed solely by the 360° scans.

due to the obstacle detection component and the latter resulted to a re-planning. Then, the robot attempted to traverse through another pair of vehicles, having adequate space. While traveling to the goal point, new obstacles are detected, such as stones or parts of vehicles that were hidden during the full circle scan. Fig. 13 illustrates a complete journey of the robot towards the end position. The gray points represent the points/obstacles detected by the laser scan which for illustrating purposes only retained their values in the heights axis. Points indicated with yellow suggest that they were both acquired by the 360° scans and used by the SVMLP component. The red colored ones correspond to obstacles yielded by the VO routine and also were used by the SVMLP. The localization of the robot is indicated by the (blue) robot and the purple squares represent the $\mathbf{p}^{i+1}$ points, dictated by the D* Lite. However, effects such as drifting lead the robot to an approximate position, which is revealed by the system. Then, the SVMLP treated this one as the current $\mathbf{p}^i$ and planned the trajectory towards the subsequent $\mathbf{p}^{i+1}$ as explained in Section 2.4. According to the VO algorithm the distance covered by the robot during our experiment was 20.35 m. The start and end locations of the robot have been measured by an external DGPS system in order to evaluate the accuracy of the VO algorithm. Due to the power supply limitations of the Era-Mobi platform continuous measurements of the DGPS were not possible. Therefore, an intuition about the accuracy of the VO algorithm after the 3D registration procedure is estimated less than 1% for 20.35 m route, i.e. 0.14 m. Summarizing, the proposed work exhibited remarkable performance and emerged the important characteristic of balancing between efficiency and safety.

Several points that need to be considered in order to avoid critical cases where the system could fail and the respective counter measures are described subsequently. A probable cause of failure can be found in-between the distance of GPP's points and the trajectory of the SVMLP, where short distances between those points in conjunction with particularly curved and "complicated" trajectories would lead to unfeasible trajectories in terms of robot's execution. The latter is compromised by considering the appropriate parameter values regarding the resolution of the GPP with respect to the robot's embodiment and the kinematics derived from the wheel constrains. Considering the platform used in our experiments, the distance between two successive points was set to 1 m, as already been mentioned in the manuscript. Additionally, the choice of the 2nd order Polynomial kernel forms trajectories with less curvature in comparison with other kernel choices, aiding the robot's task to follow such ones. Another critical case regards the distance between the virtual points and a GPP's point $\mathbf{p}^{i+1}$, which affects the robot's orientation when it reaches $\mathbf{p}^{i+1}$. The closer the virtual points to $\mathbf{p}^{i+1}$ the more aligned is the robot's orientation to that of the line segment connecting $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$. In cases where this distance is too small then the robot needs to perform agitated movements in order to follow the continuous trajectory between $\mathbf{p}^{i+1}$ and $\mathbf{p}^{i+2}$.

Regarding the environment complexity, it is considered within the SVMLP scheme, i.e. parameter T_{re} ensures the fact that the robot can traverse between them. Afterward, based on the clusters derived, each of them attains a Boolean label according to the line segment that connects $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$. Thus, a binary classification problem is formulated, where the classes are separated by a distance which is at least T_{re} that can be solved by the SVM classifier. In the case where the environment is too cluttered, i.e. the distances among the obstacles are too short, then the SVMLP dictates the GPP to replan from the robot's current position in order to find a traversable path.

4. Conclusions

The paper in hand demonstrated a local path planner operating in a closed loop fashion, for autonomous robot navigation capitalizing on the support vector theory. The proposed method inherits

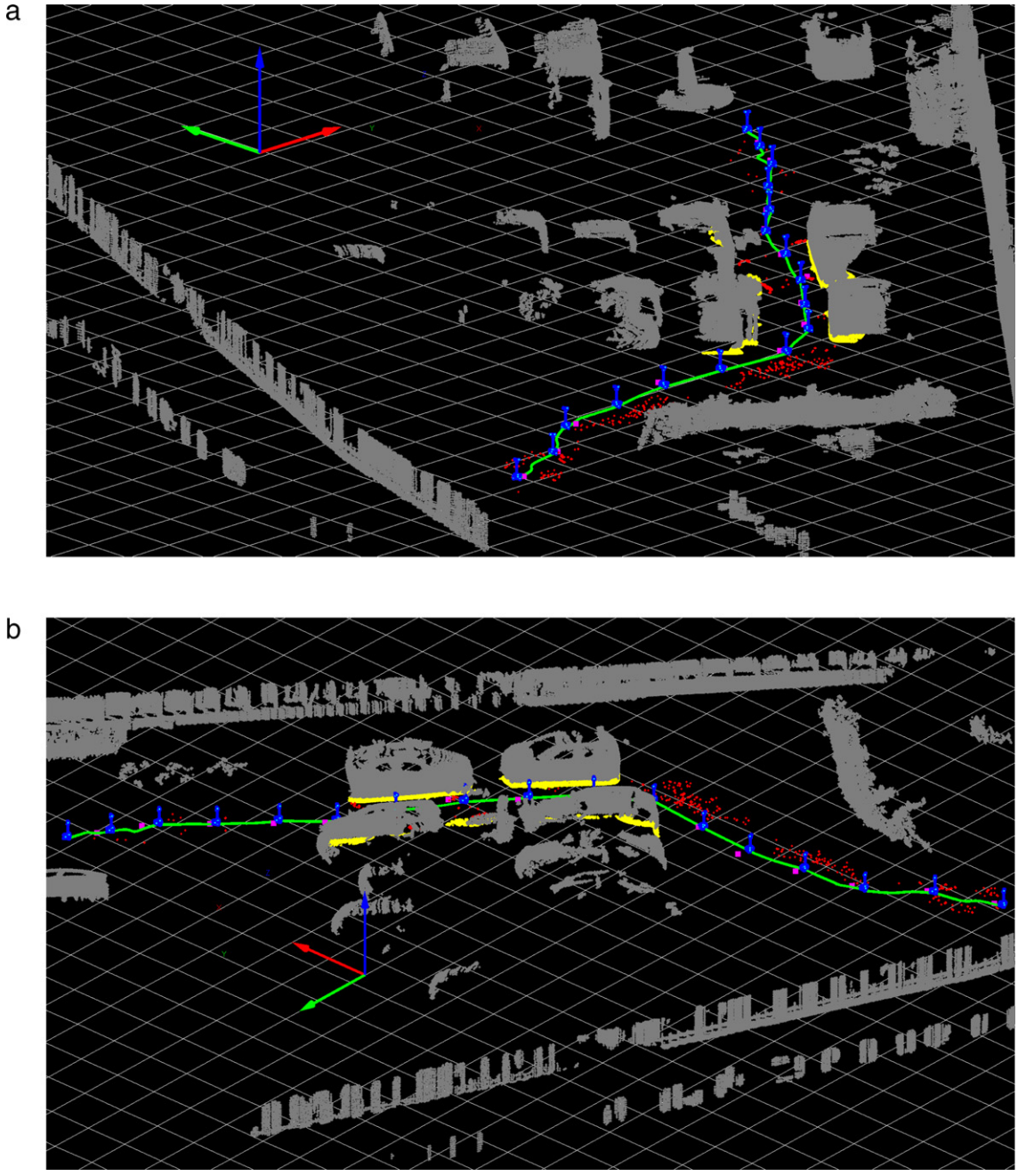


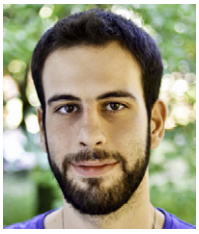
Fig. 13. Figure illustrating the trajectory of the robot in an outdoors scenario. Gray points represent the obstacles detected by the full circle scan, by means of which they retained their values in the height axis for visualization reasons only. The yellow ones are "gray points" used by the SVMLP and the red ones correspond to obstacles detected by the VO routine. The purple squares indicate the $\mathbf{p}^{i+1}$ position by the path planner, while the robot reveals its actual position. The green line indicates the continuous trajectory of the robot. (a) A viewpoint of the entire robot's trajectory once it has been concluded and (b) a second viewing angle of the robot's navigation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and combines intriguing geometrical properties, such as calculating the most secured trajectory by avoiding equally the obstacles. In order to assess the suggested local planner, it is encompassed in an autonomous navigation system, which incrementally creates the 3D metric map of the area. The registration of the point clouds is accomplished via the exploitation of the localization, the FPFH features and the ICP. The system is also capable of discovering obstacles during the robot's expedition, relying upon stereo vision and v-disparity images. The encapsulated visual odometry component provides precise estimations regarding the robot's localization. The novel SVMLP element elicits all the aforementioned components, considers the robot's embodiment and proportionally clusters as well as labels the obstacles. The acquired continuous trajectory endows all the benefits that the support vector theory offers, including the maximum margin property. The proposed work was evaluated both in separate parts and in an integrated fashion in outdoors long range scenarios. The experiments emerged all the desired properties that are theoretically enclosed in practice, offering a unique trade-off between optimality and safety.

References

[1] R.E. Kalman, et al., A new approach to linear filtering and prediction problems, J. Basic Eng. 82 (1) (1960) 35–45.
[2] S. Thrun, W. Burgard, D. Fox, et al., Probabilistic Robotics, Vol. 1, MIT Press, Cambridge, 2005.
[3] G. Welch, G. Bishop, An introduction to the Kalman filter, 1995.

[4] D. Fox, W. Burgard, F. Dellaert, S. Thrun, Monte Carlo localization: Efficient position estimation for mobile robots, in: AAAI/IAAI, 1999 (1999), pp. 343–349.
[5] F. Dellaert, D. Fox, W. Burgard, S. Thrun, Monte Carlo localization for mobile robots, in: 1999 IEEE International Conference on Robotics and Automation, 1999. Proceedings, Vol. 2, IEEE, 1999, pp. 1322–1328.
[6] S. Thrun, D. Fox, W. Burgard, et al. Monte carlo localization with mixture proposal distribution, in: AAAI/IAAI, 2000, pp. 859–865.
[7] L. Zhang, R. Zapata, P. Lépinay, Self-adaptive Monte Carlo localization for mobile robots using range sensors, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, IROS 2009, IEEE, 2009, pp. 1541–1546.
[8] M. De Berg, M. Van Kreveld, M. Overmars, O.C. Schwarzkopf, Computational Geometry, Springer, 2000.
[9] J.-C. Latombe, Robot Motion Planning, Springer, 1990, Edition en anglais.
[10] S.K. Ghosh, J.W. Burdick, A. Bhattacharya, S. Sarkar, Online algorithms with discrete visibility-exploring unknown polygonal environments, IEEE Robot. Autom. Mag. 15 (2) (2008) 67–76.
[11] A.T. Rashid, A.A. Ali, M. Frasca, L. Fortuna, Path planning with obstacle avoidance based on visibility binary tree algorithm, Robot. Auton. Syst. 61 (12) (2013) 1440–1449.
[12] S. Garrido, L. Moreno, M. Abderrahim, F. Martin, Path planning for mobile robot navigation using Voronoi diagram and fast marching, in: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2006, pp. 2376–2381.
[13] L.G. Hee, M.H. Ang Jr., An integrated algorithm for autonomous navigation of a mobile robot in an unknown environment, J. Adv. Comput. Intell. 12 (4).
[14] H. Zhang, J. Butzke, M. Likhachev, Combining global and local planning with guarantees on completeness, in: 2012 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2012, pp. 4500–4506.
[15] A. Tapus, N. Tomatis, R. Siegwart, Topological Global Localization and Mapping with Fingerprints and Uncertainty, Springer, 2006.
[16] H. Choset, K. Nagatani, Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization, IEEE Trans. Robot. Autom. 17 (2) (2001) 125–137.
[17] P. Beeson, N.K. Jong, B. Kuipers, Towards autonomous topological place detection using the extended Voronoi graph, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005, ICRA 2005, IEEE, 2005, pp. 4373–4379.
[18] Y. Koren, J. Borenstein, Potential field methods and their inherent limitations for mobile robot navigation, in: 1991 IEEE International Conference on Robotics and Automation, 1991. Proceedings, IEEE, 1991, pp. 1398–1404.
[19] J. Petereit, T. Emter, C.W. Frey, T. Kopfstedt, A. Beutel, Application of hybrid a* to an autonomous mobile robot for path planning in unstructured outdoor environments, in: Proceedings of ROBOTIK 2012; 7th German Conference on Robotics, VDE, 2012, pp. 1–6.
[20] K.M. Wurm, R. Kummerle, C. Stachniss, W. Burgard, Improving robot navigation in structured outdoor environments by identifying vegetation from laser data, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, IROS 2009, IEEE, 2009, pp. 1217–1222.
[21] J. Pan, L. Zhang, D. Manocha, Collision-free and smooth trajectory computation in cluttered environments, Int. J. Robot. Res. 31 (10) (2012) 1155–1175.
[22] J. Biswas, M. Veloso, Depth camera based localization and navigation for indoor mobile robots, in: RGB-D Workshop at RSS, 2011.
[23] M. Whitty, S. Cossell, K.S. Dang, J. Guivant, J. Katupitiya, Autonomous navigation using a real-time 3d point cloud, in: Australasian Conference on Robotics and Automation, 2010.
[24] P. Sermanet, R. Hadsell, M. Scoffier, U. Muller, Y. LeCun, Mapping and planning under uncertainty in mobile robots with long-range perception, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, IROS 2008, IEEE, 2008, pp. 2525–2530.
[25] I. Kostavelis, L. Nalpantidis, A. Gasteratos, Collision risk assessment for autonomous robots by offline traversability learning, Robot. Auton. Syst. 60 (11) (2012) 1367–1376.
[26] J. Miura, Support vector path planning, in: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2006, pp. 2894–2899.
[27] S. Sarkar, E.L. Hall, M. Kumar, Mobile Robot Path Planning Using Support Vector Machines, ASME, 2008.
[28] S. Tennety, S. Sarkar, E.L. Hall, M. Kumar, Support vector machines based mobile robot path planning in an unknown environment, Navigation 11 (2009) 12.
[29] C. Qingyang, S. Zhenping, L. Daxue, F. Yuqiang, L. Xiaohui, Local path planning for an unmanned ground vehicle based on svm, Int. J. Adv. Robot. Syst. 9 (246).
[30] R.B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (FPFH) for 3d registration, in: IEEE International Conference on Robotics and Automation, 2009, ICRA'09, IEEE, 2009, pp. 3212–3217.
[31] C. Yang, G. Medioni, Object modelling by registration of multiple range images, Image Vis. Comput. 10 (3) (1992) 145–155.
[32] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Vol. 2, Cambridge Univ. Press, 2000.
[33] S. Koenig, M. Likhachev, Fast replanning for navigation in unknown terrain, IEEE Trans. Robot. 21 (3) (2005) 354–363.
[34] S. Koenig, M. Likhachev, Incremental a*, in: NIPS, 2001, pp. 1539–1546.
[35] I. Kostavelis, E. Boukas, L. Nalpantidis, A. Gasteratos, Visual odometry for autonomous robot navigation through efficient outlier rejection, in: IEEE International Conference on Imaging Systems and Techniques, 2013, IEEE, 2013.
[36] I. Kostavelis, L. Nalpantidis, E. Boukas, M.A. Rodrigalvarez, I. Stamoulias, G. Lentaris, D. Diamantopoulos, K. Siozios, D. Soudris, A. Gasteratos, Spartan: Developing a vision system for future autonomous space exploration robots, J. Field Robot.
[37] R. Labayrade, D. Aubert, J. Tarel, Real time obstacle detection in stereovision on non flat road geometry through v-disparity representation, in: IEEE Intelligent Vehicle Symposium, 2002. Vol. 2, IEEE, 2002, pp. 646–651.



Konstantinos Charalampous was born in Thessaloniki, Greece, in 1986. He received the Bachelor's and M.Sc. (with Honors) degrees both from the School of Informatics, Aristotle University of Thessaloniki, Greece, in 2009 and 2011, respectively. He is currently a Ph.D. candidate with the Laboratory of Robotics and Automation, in the Department of Production and Management Engineering, Democritus University of Thrace. He has also participated in research projects funded by the European Commission and the Greek state. His areas of interest include machine learning and human–robot interaction.

Ioannis Kostavelis was born in Thessaloniki, Greece, in 1987. He received the diploma degree in Production and Management Engineering from the Democritus University of Thrace and the M.Sc. degree (with Honors) in Informatics from the Aristotle University of Thessaloniki in 2009 and 2011, respectively. He is currently with the Laboratory of Robotics and Automation, Department of Production and Management Engineering, Democritus University of Thrace, where he is pursuing the Ph.D. degree in the field of robotic vision. He has been involved in different research projects funded by the European Space Agency, the European Commission and the Greek state. His current research interests include vision systems for robotic applications, targeting on semantic mapping.

Antonios Gasteratos is an Associate Professor at the Department of Production and Management Engineering, Democritus University of Thrace (DUTH), Greece. He teaches the courses of Robotics, Automatic Control Systems, Measurements Technology and Electronics. He holds a B.Eng. and a Ph.D. from the Department of Electrical and Computer Engineering, DUTH, Greece. During 1999–2000 he was a visiting researcher at the Laboratory of Integrated Advanced Robotics (LIRA-Lab), DIST, University of Genoa, Italy. He has served as a reviewer to numerous Scientific Journals and International Conferences. His research interests are mainly in mechatronics and in robot vision. He has published more than 160 papers in books, journals and conferences. He is a senior member of the IEEE. More details about him are available at http://robotics.pme.duth.gr/.