A Review and Comparison of Mapping and Trajectory Selection Algorithms

Dmitrii Vershinin[1,2] and Leonid Mylnikov[1]
[1]*Perm National Research Polytechnic University, Komsomolsky avenue 29, 614990 Perm, Russian Federation*
[2]*Hof University of Applied Sciences, Alfons-Goppel-Platz-1, 95028 Hof, Germany*
nsenz@yandex.ru, leonid.mylnikov@pstu.ru

Keywords: SLAM, Scene Map, Trajectory, Cybernetics, Navigation, Autonomous Robot, Sensors.

Abstract: The dire need to solve orientation and localization tasks is directly related to the development of autonomous robotics systems as autonomous modules. In this article, we have reviewed and analyzed possible areas and peculiarities when implementing existing localization approaches in autonomous robotics systems operating under various weather conditions with possible obstacles on their way without preliminarily generated maps. In the paper, we especially pay attention to existing SLAM algorithms and a multitude of hardware concerned with this problem. Every considered and addressed algorithm in this paper comes with its main principles and generated, as a result of its performance, map type. The comparison of the algorithms was mainly based on the data of several articles and projects, in which almost perfect indoor experiments without any weather impact in order to examine the efficiency of the algorithms were conducted. Using the results acquired by the authors, a comparative table with main statistics for every considered algorithm was created. Apart from that, similar statistics for trajectory selection algorithms that meant to help researchers solve scenario/scripted tasks were covered. As a result of our review piece, we presented a ranging technique for the pair algorithms/sensors that uses the renowned TOPSIS outranking methodology. The proposed approach may become of significant help while selecting the pair for every case study.

1 INTRODUCTION

The rise of robotics dates back to the production systems efficiency problems – as a tool for assembling vehicles (cars in particular) and other complex units at factories. Today, similar robotics systems are frequently used in every aspect of human life. For example, there is a wide variety of both industrial and private cleaning robots that are also capable of scrubbing swimming pools and other surfaces, service or assistant robots and plenty of consulting robots at malls and airports. Besides, they are particularly good at helping handle with the aftermaths of technological accidents and meteorological disasters by getting to the hard-to-reach sites and seeking for casualties. Moreover, now they have become extremely popular amongst military services. For example, they can be used to collect intelligence and disarm bombs. Another fascinating use case for such robots is space in general and space exploration in particular. Not only are they used for repairing and fixing satellites, aero crafts and so on, but they also have the Moon and Mars exploration objective. Today they found their place in healthcare. Virtually they even help in agriculture and forestry. However, in order for them to be fully automotive and standalone, they need to have inside recognition, navigation and mapping algorithms. This set of algorithms and problems is commonly referred to as SLAM (Simultaneous Localization and Mapping).

These days there are too many approaches and methods for solving SLAM problems, however, only recently we have been able to notice a distinct transition to modern (hybrid) techniques and approaches, that are capable of processing data with outliers without any pre-processing, from the conventional ones (i.e. filter-based algorithms).

Additionally, in mobile robotics there is a substantial problem with processing every localization and mapping task simultaneously (S letter in SLAM) and concurrently.

Even more complex problem is when we are supposed to solve the outlined tasks in an unknown or/and unstructured and dynamically changing environment with many obstacles whether there are bump/altitude variations or sudden climate changes affecting the precision and accuracy of the sensor data.

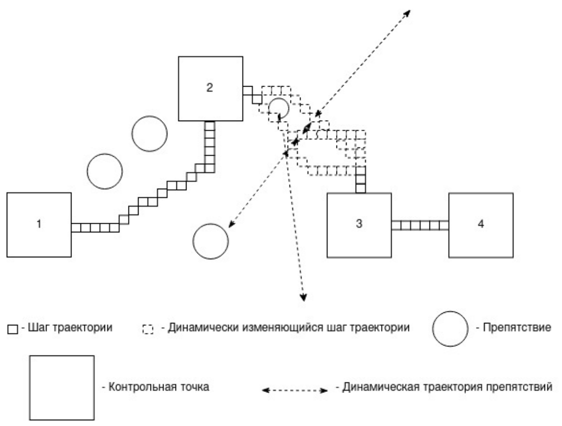
Furthermore, depending on the algorithm, an operating robot may consider using pairs of sensors of different types: acoustic, lidars, cameras, sonars, altimeters. Likewise, it is crucial to consider possible precision and accuracy requirement when opting for one or the other algorithm or sensor, since, for example, lidars are fairly efficient yet exceedingly expensive.

The type of the environment encompassing our robotics system should be considered as well, as in self-resembling and similar scenes (i.e in office rooms) the precision and accuracy requirements are likely to become stricter due to lots of outliers concerning false-positive loop closures (making the robot think that it has already visited this place).

Thus, a particular algorithm that one wants to use will be affected by the collection of sensors we have at our disposal or can use, which may constitute another worth considering and requiring a comprehensive analysis problem. And given the fact that most modern approaches are hybrid-based (it is possible to use different combinations of sensors and algorithms at run time), the mentioned selection task becomes crucial since under distinct conditions (weather, traffic, obstacles related) the efficiency of the algorithms/sensors pair may vary drastically, which is likely to lead to unpredictable results. For instance, in [1] the influence of changing weather conditions on the SLAM results of automated vehicles was shown, and in [2] authors showed how illumination and sensors positioning affect the quality of the results (i.e. the target may be behind vegetation). In [3] an autonomous adaptive multisensor SLAM was demonstrated.

2 A REVIEW OF SLAM ALGORITHMS

When comparing SLAM algorithms, the most fundamental aspect is the resulting trajectory of these algorithms. In this context let us consider trajectory as a set of points in space that are dependent on their coordinates, ambient influence on it associated with the noise of sensor data, obstacles and the positional changes of dynamic objects and obstacles. Thereby, trajectory and its possible changes are reliant on the dimensions of a robotics system. In general, a trajectory may be illustrated as a chain of interconnected coordinates, which can be seen in Figure 1.


1
2
3
4
- Шаг траектории
- Динамически изменяющийся шаг траектории
- Препятствие
- Контрольная точка
- Динамическая траектория препятствий
Figure 1: A trajectory example.

Accordingly, a trajectory step will be represented as follows: (x_i, y_i, t_i), where x_i- a robot's x position on the map, y_i- a robot's y position on the map, t_i- coordinates registration time. When there are errors/environmental influence present, the same trajectory will take the following form: $((x_i \pm c_i), (y_i \pm d_i), (t_i \pm \Delta))$, where c - external influence on the x position, d - external influence on the y position, Δ - registration deviations.

Apart from that, every SLAM algorithm generates a map of a robot's environment. This way let us represent a map as a manifold of points on the space grid containing their coordinates and probabilities of obstacles located in these areas (including the dynamic ones). But for 3D cases - a cube of grids populated with points in space. The scheme of the map is represented in Figure 2.

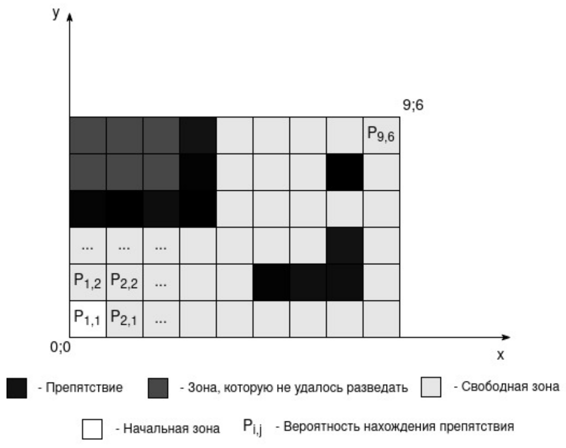


Figure 2: A schematic representation of an environmental map.

In this case P – the probability indicating a present obstacle in a particular cell of the grid. The cells that have been visited by the robot are coloured light-grey, dark-grey – for uncharted cells and black - for walls or static obstacles.

2.1 SLAM Algorithms

Hector SLAM – is a SLAM method that operates by means of extracting data from a 2D lidar. At the moment, it is one of the most popular approaches that is, on top of that, widely used in a variety of mobile robotics projects. The algorithm builds a 2D map and provides localization possibilities at the scan rate of the lidar. In order to build a correct map, a conversion from the local lidar‘s coordinate system to the surface, that the robot is moving through, coordinate system should be performed [4].

Hector SLAM builds an occupancy grid (the map that corresponds to the one described in our definition), in which every cell is coloured: black – the cell is occupied, light-grey – the cell is empty, dark-grey – the cell has not been checked yet. An example of the resulting map can be seen in Figure 3.

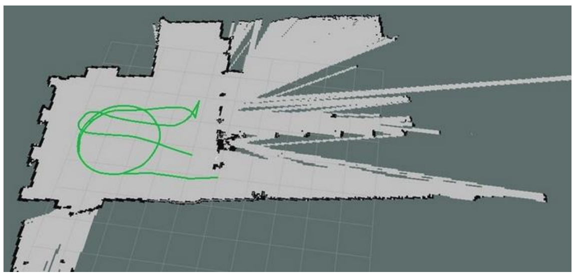


Figure 3: A Hector SLAM resulting map [5].

ORB-SLAM – is a versatile and accurate SLAM algorithm, based on features recognition and real-time trajectory calculation utilizing monocular cameras, which builds an environment sparse 3D scene map. It can close large loops and perform global relocalisation in real-time and from wide baselines. Apart from that, ORB-SLAM makes it possible to automatically initialize scenes of different types [6]. The resulting map of this algorithm is a sparse 3D map, an example of which is illustrated in Figure 4.

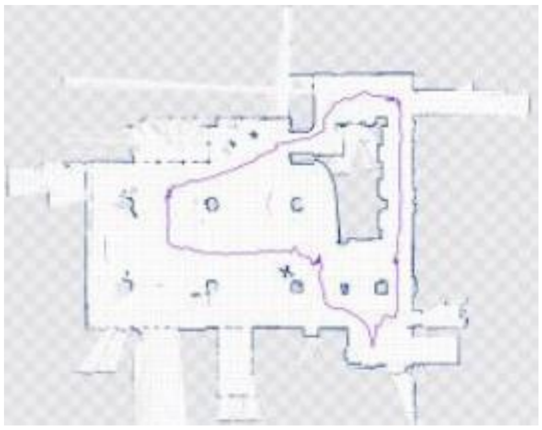


Figure 4: An ORB-SLAM resulting map [7].

DPPTAM – is one of the newest visual SLAM algorithms, which adjusted and implemented the most successful ideas of the previous algorithms. It is a direct monocular odometry algorithm that estimates a dense reconstruction of a scene in real-time on CPU and saves the trajectory as a sequence of points in the particles cloud. To build high-resolution images the algorithm makes use of standard techniques for minimizing the points errors [8]. An example of this map is shown in Figure 5.

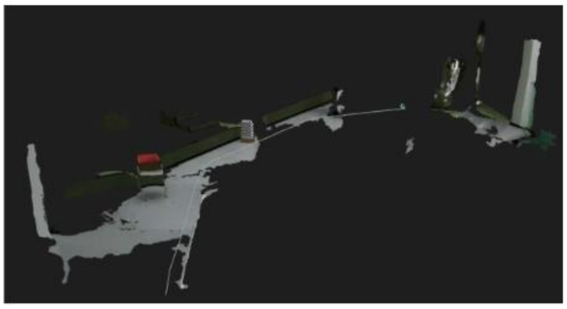


Figure 5: A DPPTAM resulting map [9].

ZEDfu – tracks positioning and orientation based on a ZED camera mounted on a tracking device. A ZED camera builds a real-time 3D world and recognizes rooms and objects. As a resulting map, the algorithm builds a 3D lattice from particles clouds of any environment (either indoors or outdoors) [10]. An example of this map can be seen in Figure 6.

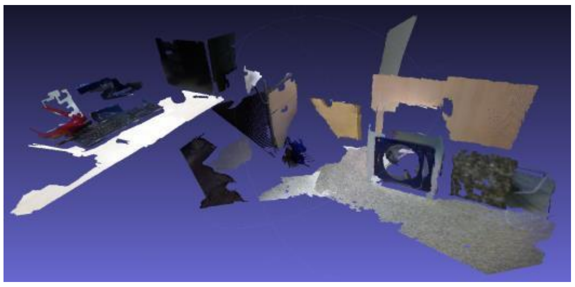


Figure 6: A ZEDfu resulting map [10].

RTAB-Map – is a RGB-D, Stereo and Lidar Graph-Based SLAM approach based on an incremental appearance-based loop closure detector. The loop closure detector uses a bag-of-words approach to determinate how likely it is that a new image comes from a previous location or a new location. When a loop closure hypothesis is accepted, a new constraint is added to the map's graph, then a graph optimizer minimizes the errors in the map. RTAB-Map can be used either with a Kinect or with a stereo camera/lidar [11]. An example of this map is shown in Figure 7.

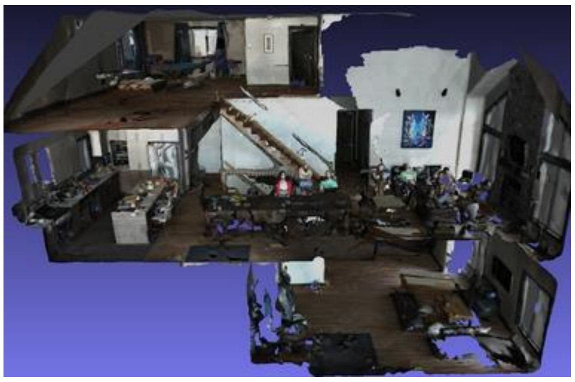


Figure 7: An RTAB-Map resulting map [11].

Using such SLAM methods, we can build an environmental map and localize the considered object in space, however, in order to solve possible scenario tasks, perform various operations (i.e. holding or raising objects, approaching them in the most efficient way and performing manipulations with them) we may need to use proper trajectory selection algorithms.

An overall comparison of the algorithms' characteristics can be seen in Table 1.

2.2 Trajectory Multi-Objective Optimization Algorithms

MOACO is a multi-group trajectory optimization algorithm. MOCAO uses multiple pheromone matrices and more heuristic matrices. Each of these matrices is responsible for only one task. All the agents are divided into several groups. Each group has multiple weights and each agent in the group has its weight vector. If the number of weights in the group is less than the number of agents, then the other agents are set weights from the beginning. Thus, two or more agents in a group may use the same weight vector. But every agent uses its vector to aggregate the pheromone and heuristic information. Afterwards, it calculates its probability to move to an unvisited spot and chooses the next spot to visit via wheel roulette selection [13]. Finally, it uses non-dominated solution generated by current iteration to update the pheromone information.

MACS uses one pheromone matrix and multiple heuristic matrices. Each heuristic matrix is responsible for only one task. Each agent has a weight vector and all the heuristic matrices are aggregated by weighted product. The weight vectors between two distinct agents are different and non-dominant solution is used to update the pheromone information. MACS considered to be a similar to MOAQ approach with only one difference in the number of the weight vectors used to aggregate heuristic information: in MOAQ two such vectors are used {0,1} and {1,0}, whereas MACS uses more vectors owing to its influence on the algorithm [13].

PACO algorithm uses multiple pheromone matrices and only one heuristic matrix. All the agents share this single matrix. Each pheromone matrix is responsible for only one task. As in MACS algorithm, each agent has its weight vector and all the pheromone matrices are aggregated by the weighted sum. This algorithm uses the best option and the second to the best solution of each objective to update pheromone information. The non-dominated solutions mainly approximate to the central part of the Pareto front [13].

MOEA is one of the most efficient agent algorithms. The algorithm performs multi-object decompositions and simultaneously optimizes its parts. Each subtask is optimized by using the nearby subtasks information [13].

The algorithms comparison is presented in Table 2.

Table 1: SLAM Algorithms comparison (based on [12] data).

Algorithm	Accuracy	Algorithm type	Quality of the map	Best suited for	Odometry quality	Max deviation (м)	Obstacles avoidance precision	RMSE(м)	Mean (м)	Std (м)
Hector	High	Occupancy grid	Good	2D lidar	–	0.18	Excellent	0.088	0.025	0.024
ORB	High	Feature based map	Low	Monocular camera	Good (0.43м)	0.43	Good	0.166	0.159	0.047
DPPTAM	Average	Cloud of particles based map	Average	Monocular camera	Bad (4.26м)	4.26	Good	0.338	0.268	0.206
ZEDfu	Very high	Cloud of particles based map	Good	Stereo ZED camera/Kinect	Good (0.32м)	0.32	Good	0.726	0.631	0.358
RTAB	Above average	Cloud of particles based map	Good	Kinect	Good (0.67м)	0.67	Average	0.163	0.138	0.085

Table 2: Trajectory multi-objective optimization algorithms comparison (based on [13] data).

Algorithm	Max (м)	Min (м)	Mean (м)	Std (м)
MOACO (kroAB100)	0.3412	0.3037	0.3236	0.0105447
MACS (kroAB100)	0.1924	0.1675	0.1823	0.0054743
PACO (kroAB100)	0.4076	0.3695	0.3912	0.011176
MOEA (kroAB100)	0.1062	0.0505	0.0767	0.0144815
MOACO (kroAC100)	0.3475	0.3211	0.3352	0.0063372
MACS (kroAC100)	0.1995	0.1723	0.1885	0.0051235
PACO (kroAC100)	0.2612	0.2285	0.2413	0.0086281

3 HARDWARE USED BY THE ALGORITHMS

As it has been mentioned before, the selection of a sensor for a particular algorithm is an essential task. It is mainly due to the technical characteristics of the sensors and their use case limitations. Thus in this section, an excerpt from a paper on sensors efficiency when a variety of obstacles is present will be presented.

First of all, the most conventional robotics sensor types should be enumerated and described here:

- Proximity sensors – detect objects that are located in close proximity to the robot. These sensors can detect objects' presence by using light, sound or electromagnetic fields (for example, infrared, ultrasound sensors and LDRs).
- Rangefinders – determine the distance between two distinct objects in an environment (for example, cameras, lasers, lidars).
- Tactile sensors – provide information about physical contacts with objects.
- Light sensors – detect light density that consequently can be converted into current or voltage.
- Sound sensors – detect sound and return proportional to the sound level voltage.
- V/I Converters.

Let us consider the most common sensors for detecting objects and obstacles.

RADAR/LIDAR. To detect obstacles the RADAR (radio detection and ranging) /LIDAR (light detection and ranging) combination is frequently used.

Detection and distance measurements are one of the main LIDAR functions. The distance is represented as the time required for a light impulse to travel from a sender to a photodetector after its reflection from an object/obstacle surface. The distance is defined as: , where – represents the distance, – the speed of light, – the impulse time. Therefore, LIDAR can obtain objects' 3D geometry [12] and [14].

Cameras are probably the most popular sensors used to detect objects and environmental changes.

Their main peculiarity is in the ability to recreate a 3D cloud of particles of a particular environment.

Table 3: Sensors' characteristics comparison (based on [14] data).

Sensor's type	Size	Power consumption (Вт)	Depth (м)	Price ($)	Effective precision	Best suited for	Acceptable temperature range	Requires additional hardware	Dependent on illumination	How representative for people	Efficiency under bad weather conditions
Acoustic	Very compact	0.01 - 1	2-5	10-500	1-3% of max depth	Range/acoustic based SLAM	From -20 Up to +80	-	-	Bad	Weather resistant but operate poorly in noisy environments
Monocular cameras	Very compact	0.01 - 10	-	100-5k	25–135 mm	Visual SLAM	Depends on a camera, however in general these cameras have a pretty decent temperature resistance	+	+	Good	Weather resistant provided the lenses are clean
Lidar	Bulky	50 - 200	50-300	5k-100k	Up to ± 3 cm	Range/distance based SLAM	From -50 Up to +80	-	-	Bad	Efficiency drops in rainy, snowy and foggy conditions
Stereo cameras	Compact	2 - 15	5-20	500-5k	1mm - 5см	Visual SLAM	-	+	+	Good	Weather resistant provided the lenses are clean

Apart from that, cameras are able to detect obstacles by creating depth maps from consecutive images fetched by monocular cameras. However, this approach works perfectly with only static obstacles [12], [14].

SONAR (Sound navigation and ranging) – creates a sound impulse and measures the impulse's echo return time. Therefore, the sensor's results cannot be affected by light or illumination. However, sonars are mainly used for nearby detections, meaning that they are futile when it comes to measuring distant objects. But, an even worse drawback of the type is in its inability to operate in noisy environments (engine vibrations, highways, toots) as was shown in [12] and [14].

Laser rangefinder. The measurement principle is based on the angle between the laser ray pointing at an object and the laser's lens. Having this laser-lens distance (h) and the angle, we can calculate the distance to the object – the less the angle, the farther the object [12] and [14].

The comparison of the sensors' characteristics is represented in Table 3.

The sensors that can operate under well-illuminated conditions may easily avoid smoke conditions or even mist. However, when it comes to cameras, it is important to make them able to use infrared or thermal vision in order to increase their performance under such conditions.

Under rainy conditions, the most efficient sensors are LIDARs, Laser rangefinders and some types of cameras.

When operating in a blizzard, the most efficient sensors are cameras, since LIDAR/RADAR systems may be covered with snow, which prevents them from delivering any acceptable result.

Working in a highly reflective environment, neither of the sensors without supportive filtering algorithms demonstrated their efficiency.

When there are physical obstacles present (slopes, hills, slides etc.) the most efficient systems are LIDAR and multi-camera systems [1].

Owing to combinations of monocular cameras, it becomes possible to detect a wide range of obstacles around our object. However, in comparison with the cameras, LIDARs provide much better precision and FOV.

However, sometimes LIDAR data becomes incorrect due-to its distance to an object/obstacle. As in the cameras' case, some of the LIDAR sensors may be exceedingly noise sensitive.

Therefore, in a wide variety of modern robotics systems, researches use hybrid approaches. For example, some of them propose a combination of 6 SONARs with 3 Visual cameras for obstacles detection under any condition. Besides, it is a good practice to use LIDARs and RADARs together to reduce the resulting errors. In order to boost the performance even further, it is helpful to add visual cameras to the aforementioned combination to be able to obtain information about roads, road signs, signals, etc [1].

So, we can conclude that these hybrid systems, which use a set of SLAM algorithms, are becoming more and more popular and relevant due to the importance of changing of adapting SLAM algorithms at runtime.

4 RESULTS

As a result of the review, we can formulate a trajectory/SLAM selection algorithm that will make it easier for researches to opt for a particular set of sensors/algorithms and take into account existing restrictions and constraints (see Figure 8).

Step 1: Define the robotics systems restrictions and constraints (financial, technical) and the system requirements (ability to operate under different weather conditions, avoid obstacles, operate in noisy environments etc.).
Step 2: Formulate a list of parameters/measurements from the restrictions and requirements.
Step 3: Perform the ranking operation of the solutions (individually for algorithms and hardware) by pairwise considerations of the parameters/measurements as shown in Figure 9.
Step 4: Based on the ranging results the expert should choose a pair(s) of the algorithm(s)/sensor(s) depending on their priorities and preferences. The initial selection of several pairs will sift the range of available options making it easier for them to expertly select the most suitable option.

Figure 8: The selection algorithm for choosing the most suitable methods and sensors for robotics systems operating in any environment.

To perform the third step one may make use of some of the following outranking methods: TOPSIS [15], ELECTRE, VIKOR, PROMETHE. For example, in Figure 9 the ranking criteria are represented as x and y axes, points represent the selected algorithms from the previous steps. To include a researcher's subjective point of view, a particular sign is plotted on this chart (in this case, it is a diamond). Then, the same ranking approach should be performed for the remaining parameters/measurements. Based on the distance from the best and the worst option the researcher will be able to select the most suitable algorithms.

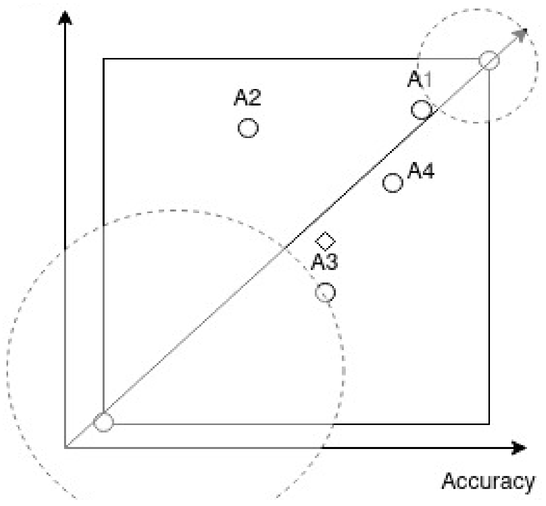


Figure 9: A ranking example (A1 - Hector SLAM, A2 - DPPTAM SLAM, A3 - ORB SLAM, A4 - RTAB SLAM, prices were set as approximate based on the previous tables).

5 CONCLUSIONS

There are no comprehensive robotics systems that are able to operate under any potential condition. Therefore, based on a detailed analysis (like this one), we can select the most appropriate set of algorithms/sensors for our particular case to operate under required conditions (weather, obstacles related). However, it is crucially important to understand that apart from the covered conditions, there might be some additional ones (for example, related to financing the project, difficulty of the project, personal preferences, the experience of the team and so on) and, in this case, the eventual choices may vary. Therefore, this paper cannot solve or cover all the problems regarding that selection, but this review may still be considered as a useful handbook.

According to this article we may conclude that at the moment there are two approaches to solving SLAM problems regarding the algorithmic part: 1) Based on our requirements, we can create new algorithms and systems by combining the existing ones and 2) Develop brand new algorithms that would entirely solve our problems based on the requirements and limitations.

REFERENCES

[1] X. Yu and M. Marinov, “A study on recent developments and issues with obstacle detection systems for automated vehicles,” Sustain., vol. 12, no. 8, 2020, doi: 10.3390/SU12083281.

[2] P. Slivitsin, A. Bachurin, and L. Mylnikov, “Robotic system position control algorithm based on target object recognition,” Proc. Int. Conf. Appl. Innov. It, vol. 8, no. 1, pp. 87-94, 2020.

[3] Sh. Shen, “Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft micro-aerial vehicle (MAV),” US10732647B2, 2013.

[4] S. Kohlbrecher and J. Meyer, “Hector SLAM” [Online]. Available: http://wiki.ros.org/hector_slam, 2012.

[5] W. A. S. Norzam, H. F. Hawari, and K. Kamarudin, “Analysis of Mobile Robot Indoor Mapping using GMapping Based SLAM with Different Parameter,” IOP Conf. Ser. Mater. Sci. Eng., vol. 705, no. 1, 2019, doi: 10.1088/1757-899X/705/1/012037.

[6] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: A Versatile and Accurate Monocular SLAM System,” IEEE Trans. Robot., vol. 31, no. 5, pp. 1147-1163, 2015, doi: 10.1109/TRO.2015. 2463671.

[7] M. Sokolov, O. Bulichev, and I. Afanasyev, “Analysis of ROS-based visual and lidar odometry for a teleoperated crawler-type robot in indoor environment,” ICINCO 2017 - Proc. 14th Int. Conf. Informatics Control. Autom. Robot., vol. 2, no. July, pp. 316-321, 2017, doi: 10.5220/0006420603160321.

[8] A. Concha and J. Civera, “DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence,” IEEE Int. Conf. Intell. Robot. Syst., vol. 2015-December, no. July, pp. 5686-5693, 2015, doi: 10.1109/IROS.2015.7354184.

[9] StereoLABS, “ZEDfu” [Online]. Available: https://www.stereolabs.com/docs/.

[10] I. Z. Ibragimov and I. M. Afanasyev, “Comparison of ROS-based visual SLAM methods in homogeneous indoor environment,” 2017 14th Work. Positioning, Navig. Commun. WPNC 2017, vol. 2018-January, no. October, pp. 1-6, 2018, doi: 10.1109/WPNC.2017. 8250081.

[11] M. Labbé and F.¸Michaud, “RTAB-Map as an Open-Source Lidar and Visual SLAM Library for Large-Scale and Long-Term Online Operation,” J. F. Robot., vol. 36, pp. 416-446, 2019.

[12] M. Filipenko and I. Afanasyev, “Comparison of Various SLAM Systems for Mobile Robot in an Indoor Environment,” 9th Int. Conf. Intell. Syst. 2018 Theory, Res. Innov. Appl. IS 2018 - Proc., no. November, pp. 400-407, 2018, doi: 10.1109/IS.2018.8710464.

[13] J. Ning, C. Zhang, P. Sun, and Y. Feng, “Comparative study of ant colony algorithms for multi-objective optimization,” Inf., vol. 10, no. 1, pp. 1-19, 2018, doi: 10.3390/info10010011.

[14] M. Zaffar, S. Ehsan, R. Stolkin, and K. M. D. Maier, “Sensors, SLAM and Long-term Autonomy: A Review,” 2018 NASA/ESA Conf. Adapt. Hardw. Syst. AHS 2018, pp. 285-290, 2018, doi: 10.1109/AHS.2018.8541483.

[15] Z. Pavić and V. Novoselac, “Notes on TOPSIS Method,” Int. J. Res. Eng. Sci., vol. 1, no. 2 [Online]. Available: https://www.researchgate.net/ publication/ 285886027_Notes_on_TOPSIS_Metho%0Awww.ijres.org, pp. 5-12, 2013.