Control movement of mobile robots inside building based on pattern recognition algorithm

I.A. Shcherbatov
Department of Automation and Control
Astrakhan State Technical University
Astrakhan, Russia
sherbatov2004@mail.ru

Nguyen Tuan Dung
Viet Nam Academy of Science and Technology
Ha Noi, Viet Nam
dunghvkt@yahoo.com

V.P. Glazkov
Department of Applied Information Technologies
Yuri Gagarin State Technical University of Saratov
Saratov, Russia
robsii@sstu.ru

O.M Protalinskiy
Department of Applied Informatics in Economics
Astrakhan State Technical University
Astrakhan, Russia
protalinskiy@gmail.com

Abstract—in the movement management system of mobile robots, simultaneous detection of obstacles and creation of obstacles’ map in the near field of the mobile robot are extremely urgent and complex tasks. In general, environment around robot is very complicated due to differences in change of lighting conditions and viewing angles. These problems lead to decrease accuracy of obstacles recognition. In this paper, we propose algorithms to solve this problem and increase effectiveness of mobile robots: to detect obstacles and create map of environment around robots could be used algorithm, which based on 3D-point cloud. In addition, algorithm “SURF” is implemented to detect landmark. Using this algorithm allows satisfying requirements for obstacle detection in real time. The experimental results show the effectiveness of proposed approach using in vision system of a mobile robot platform.

Keywords—Kinect, depth image, RGB image, 3D-point cloud, 2D point, SURF, FLANN, mobile robot, image recognition

I. INTRODUCTION

The main objective of movement management of mobile robots is the solution to remove the obstacles around robots. Mobile robots have used various navigation systems to control the movements, which based on sensors or navigation signal (for example, stereo cameras [16, 17] Electronic compass [6], inclinometer [6], sonar [6], etc.). The functionality of these approaches depends on the success of tracking algorithms. For this purpose, two methods obstacle detection are commonly used: Object recognition methods based on RGB - images (Method 1) and the spatial-temporal approach (method 2). The first class of methods in many cases leads to a decrease in recognition accuracy and computational power due to differences in the change of lighting conditions and viewing angles. To eliminate these drawbacks second class of methods is used to ensure that obstacles are detected with their specific areas of the image.

In navigation of mobile robot, path-planning algorithms are designed to obtain an optimal path between the starting point and the destination without colliding with obstacles [2, 9]. Typically, path planning is divided into linear segments. Robot will move on the path, which based on these line segments [6].

In addition, to calculate the shortest safe paths several methods are used in a local or global motion planning [16]. Global path planning takes into account all information of environment. For this purpose, several methods have been used: Voronoi diagram [7, 13], cell decomposition [18], etc. The implementation of these approaches requires a considerable time in pre-calculation and auxiliary hardware to increase performance [7]. The main disadvantage: they need to create a database of images of the obstacles surrounding area to build its topology and connectivity facilities.

Local path planning is designed to prevent collision with obstacles near the robot. It uses only information about nearby obstacles. The best known are techniques based on potential fields [19]. It divided the space into a regular grid, on which we can search for an optimal path without colliding with obstacles. These methods are quite quick, but do not allow preventing an infinite loop (local minima of the potential function). To eliminate the problem of local minima and obtain an optimal way, the approach described in [17] could be used.

II. PROBLEM STATEMENT

When control the movement of mobile robots in unknown environment, appear some difficulties in obstacles detection and the using of motion path planning algorithms. To eliminate them, we need to develop a method that uses information, which is based mainly on the depth image, for example, data is
obtained from the Kinect sensor and a navigation algorithm, which help robot to move without collisions. To do this, required to solve three interrelated subtasks:

- Implement object detection algorithm from RGB - images for landmark detection
- Propose method for building obstacles map from depth image
- Propose an algorithm for movement management of robot.

III. GENERAL SYSTEM ALGORITHM

Kinect is a new camera system [3]. It has a number of advantages over traditional sensors, which can work in low light levels and it is invariant with respect to color and texture. The resulting images have a normal depth of 640x480 pixels at 30 frames per second. Therefore, our system can works with all type of images from Kinect. The basic structure of system is shown in Fig. 1:

From Figure 1, it is clearly that the method consists of three major steps: obstacle detection (landmark and other obstacles detection), creation of obstacles map, path planning.

A. Obstacle detection

1) Landmark detection

Object detection from image or video stream is challenging problem. Because the differences in the occupied position, color, viewing angles, illumination conditions. In recent years, a considerable number of studies in this field have proposed various methods. Most researches are based on the images that obtained from the visible spectrum in a similar way, as the human eye perceives it. Some methods use statistical sampling, which based on local features (e.g., HOG - Histograms of Oriented Gradients [10], EOH - Edge orientation histograms [10]), and other base on extraction of singular points from image (e.g., SIFT -Scale Invariant Feature Transform [12], SURF - Speeded Up Robust Features [1]). These methods allow obtaining accurate results for the detection of objects.

From our researches [3, 4, 5] and other [1; 12], we propose a method, which uses nearest neighbor search «FLANN - fast library for approximate nearest neighbors», and algorithm «SURF» to find matching images in the collection of images (figure 2.). Library «FLANN» contains a collection of algorithms: extracting descriptors of the image, computing descriptors for multiple images, create index FLANN, performing searches KNN (k-nearest neighbor method), and getting results. Efficiency of finding objects will increase with the use of the algorithm SURF. Using library FLANN conjunction with SURF algorithm meets the requirements for detection of objects in real time. The experimental results (Fig.2) show the effectiveness of the proposed approach when used in the vision system of mobile robot platform

2) Other obstacles detection

Various objects detection in images or video stream is a difficult task. Because they have many differences in the occupied positions, colors, viewing angles, lighting conditions, etc. In recent years, a considerable number of researchs in this field propose various methods for their detection. However, they have some disadvantage: they encountered problems when recognize people or change in image backgrounds. This leads to decrease recognition accuracy, increase power computing, and hence the time of object recognition.

Depth information can be used to eliminate these disadvantages and allow identifying objects. Any objects may have consistent color and texture, but should occupy a certain area in the space. Depth images have several advantages over 2D-images: they are resistant to change color and lighting; they are simple representations of 3D-information.

3D-point cloud is created from depth data, which obtained from Kinect [14]. Ki-nect uses infrared camera to create depth image. 3D – points cloud are defined as a set of unorganized irregular points in 3D. They are not only save coordinates of all points on reflected surface of obstacles, but also provide additional information of these points (for example, intensity, and RGB values, etc.).
The effectiveness of obstacle detection, which use 3D-point cloud, was proved in [4, 14]. This method can be used in real time to detect moving and stationary objects. The basic structure of algorithm is shown in figure 3.

The effectiveness of this algorithm is higher than other traditional algorithms, which are based on images obtained from the visible spectrum.

B. The creation of obstacles map and building trajectories

In general, the environment around robot is very complicated due to the differences in changes in lighting conditions and viewing angles and obstacles. Differences can change quickly. These problems lead to a decrease in precision of motion control of the robot. Therefore, map creation is a key step.

Robot need to know all main information about the obstacles to move without colliding (e.g., current location, size of the obstacles, etc.). Illustration of obstacle map is shown in Figure 4.

The algorithm consists of three sub-tasks: obtaining information about obstacles, calculating the distances from «Kinect» to the obstacles and the shortest distance between obstacles, obtaining all required points. The main procedure of the algorithm has the following steps:

1. Obtaining the borders the obstacles. These borders could be concave hull or convex hull.
2. Calculate all the distances from Kinect to all obstacles that are sorted in order from closest to farthest to the robot. In addition, the list of obstacles will be saved.
3. Calculate distances from any obstacle to the rest of obstacles. These distances will be saved in array DI.
4. Find midpoint of lines, which connect two obstacles in array DI. If length of this line is higher than a minimum acceptable value (MAV), then this midpoint is acceptable, otherwise is unacceptable. From figure 4, shown that all the points {A, B, C, D, F} are acceptable and point E is unacceptable. All valid points will be saved in array AVP.

This array is input data of next step “Path planning”.

C. Path planning.

1) Create pre-calculate trajectory

In the step “building trajectory”, robot finds the minimal path from itself to the goal. It means that robots can move in the space with a minimum cost of path from any point to the goal. Array AVP will be used as input data for calculation of movement path. The set of points in AVP is defined:

$$P = \{P_1, P_2, ..., P_k, ..., P_n\}$$

Where, $$P_k = (x_k, y_k)$$ is adjacent points. In P, no point is repeated. The last point is the goal. Algorithm starts at the first point $$P_1$$ and search the best path with minimal cost.

In general, the cost of path is an arbitrary function of a discrete way $$F(P)$$. We assume that the cost of the path is divided into the sum of the he intrinsic value of the point with the cost of moving from current point to the next:

$$F(P) = \sum_i G(p_i) + \sum_{i+1} H(p_i, p_{i+1})$$

Both G and H are arbitrary functions. With typical situations, G represent the cost of the path through the specified point to the current point $$P_i$$. H is Euclidean distance, which robot moves between two points $$P_i$$ and $$P_{i+1}$$.

For this purpose, some popular methods could be implemented, such as A* - A star, algorithm Dijkstra, D*, hill climbing … but algorithm A* is the most popular.

The algorithm A* is used to find the path traversing of a graph or in the process of drawing up an effective plan between points. In the general case, the algorithm finds a path from starting to destination node. The heuristic function is used to classify each unit in accordance with the evaluation of the optimal route through any node.

The algorithm scans the entire path from the initial vertex to the end in order to find the minimum cost. At each stage, handled the set of paths from the starting node to all the undiscovered the vertices which is located in a queue with priority determined by a value

$$f(n) = g(n) + h(n)$$

Where,

$$g(n) = \sum_{i=1}^{n} d(i)$$
\[ d(i) = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \]  
(2)

And \[ h(n) = \sqrt{(x_n - x_0)^2 + (y_n - y_0)^2} \]  
(3)

Where \( g(n) \) – the cost of path from the start point to the point \( n \), \( h(n) \) - the distance to the target along a straight line.

Algorithm stops when the value \( f(x) \) of target vertex is minimal. This means a solution will be chosen with the lowest cost. The less \( h(x) \), the higher the priority. Therefore, sorting trees have been used for the implementation of the queue [15]

2) Prevent collisions with obstacles

After building the obstacles maps of environment and creating pre-calculate trajectory, the robot begins to move when data is received from Kinect. These data are continuously transformed into 3D - point cloud. If the trajectory intersects with any new obstacles or the distance to the obstacle is less than the acceptable minimum value, the robot stops. After that, the current 3D - points cloud are added to the map of obstacles and robot will create a new map of the environment

Environmental conditions for performing experiments: the robot moves inside the building, the parameters of the environment: the minimum distance from the robot - 0.4 m; maximum distance from the robot - 8 m, height - 2 m; in this building have some obstacles (movement and stationary objects).

The experiments were performed to detect obstacles using the point cloud and algorithm “SURF”. The results presented in Fig. 6 and table 1.

![Flowchart of preventing collision with new obstacle](image)

After receiving a new map of obstacles, the algorithm calculates another short safe path from the current position of the robot to the goal. However, the previous path is maintained, and if the obstacles blocking movement disappear, the robot continues to move in accordance with the previous path. In other words, the algorithm is independent of dynamic changes in the environment (fig.5.)

IV. EXPERIMENTS AND RESULTS ANALYSIS

The verification of our experimental was performed on a personal computer with the following specifications: CPU Intel Pentium Duo T2390 with a frequency 1.87GHz, 1GB of RAM, 256MB of video memory (computers with these characteristics can be installed on mobile robotic platform) [17, 18].

![Results of the detection algorithm using «3D- point cloud»](image)

![Specific times of 3 main steps](image)

<table>
<thead>
<tr>
<th>TABLE I. RESULTS OF EXPERIMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>distance, m</td>
</tr>
<tr>
<td>obstacle</td>
</tr>
<tr>
<td>Object detection time - ODT (1)</td>
</tr>
<tr>
<td>Time to create obstacle map - TCM (2)</td>
</tr>
<tr>
<td>Time to create trajectory - TCT (3)</td>
</tr>
<tr>
<td>total time - TT (1+2+3)</td>
</tr>
</tbody>
</table>

From Fig. 6 and Table. 1 shows that the performance of the algorithm is the high-est, if the maximum distance to the obstacle is less than 2.5 m. Time of movement control of the robot is divided into three main parts (Table. 1): Object detection time – ODT; Time to create obstacle map - TCM;
Time to create trajectory – TCT. However, Object detection time is the most principal:

\[ TT = ODT + TCM + TCT \]  \hspace{1cm} (4)

These experiments show that the construction of the trajectory of the robot should be used algorithm, which base on 3D – point cloud, to detect obstacles and create obstacles map and for the construction of the trajectory of the robot can use the algorithm "A **".

V. CONCLUSION

These experiments show that the obstacles detection should be performed by using an algorithm with Point cloud. With the using of this algorithm together with algorithm A* allow to build movement trajectory of robots in unknown environments. The effectiveness of this algorithm is higher than traditional algorithms. In addition, algorithm “SURF” can be used to detect landmark.

The algorithms can be performed on any other range finders, stereo cameras, which return colored frames, depth images.

REFERENCES