Method of mobile robot indoor navigation by artificial landmarks with use of computer vision

E S Glibin, A A Shevtsov, O A Enik

Togliatti State University, 14 Belorussskaya St., Togliatti, 445020 Russia

E-mail: zhenya118@gmail.com

Abstract. The article describes an algorithm of the mobile robot indoor navigation based on the use of visual odometry. The results of the experiment identifying calculation errors in the distance traveled on a slip are presented. It is shown that the use of computer vision allows one to correct erroneous coordinates of the robot with the help of artificial landmarks. The control system utilizing the proposed method has been realized on the basis of Arduino Mego 2560 controller and a single-board computer Raspberry Pi 3. The results of the experiment on the mobile robot navigation with the use of this control system are presented.

1. Introduction
One of the mobile robots’ tasks is the indoor movement according to previously known trajectories, for example, employing the robots for autonomous transportation of parts or goods around manufacturing and storage rooms or as advertising guides. At the same time, there is the problem of the autonomous mobile robots indoor navigation due to the inability to use satellite navigation systems. In the specialized literature, research papers dedicated to indoor navigation systems are widely presented, among which the navigation systems using Bluetooth and ultrasonic beacons, Wi-Fi signals, black lines, etc. can be highlighted [1, 2]. But all of them have serious limitations.

The solutions focused on the odometry method that uses wheel movement data to evaluate the robot position are of particular interest [3]. For the autonomous indoor movement of the wheel robot, it is possible to calculate the distance traveled by each wheel per unit of time based on the wheel size using the angular motion sensors or encoders mounted on the shafts of the wheel motors.

Table 1. Error of the robot positioning estimation using the odometry method

| Surface material | Trajectory type          | Mean deviation, cm |
|------------------|--------------------------|--------------------|
| linoleum         | straight line 2 m        | 5.2                |
| linoleum         | right angle 2 m x 1.5 m  | 8.2                |
| tiles            | straight line 2 m        | 6.1                |
| tiles            | right angle 2 m x 1.5 m  | 8.5                |
| tiles            | straight line 20 m       | 54                 |

Integrating the speed of wheels rotation, one can constantly monitor the coordinates of the robot. However, due to wheel slippage, the error of the coordinate determination will inevitably increase in...
the future. This will be particularly noticeable when the robot turns or when it moves over an uneven surface. Table 1 shows the experimental results of a three-wheeled robot motion on the surfaces of different types. The distance between the leading wheels of the experienced robot was 350 mm, and the wheel diameter was 120 mm.

Thus, it is necessary to develop a navigation method that provides high accuracy in estimating the position of the mobile robot indoors.

2. Solution approach
To solve the problem described above, a technology of computer vision is used [4-6]. Coordinates can be adjusted if an artificial landmark is located in a room at a known place (Figure 1). Having fixed this landmark with a video camera, the robot determines its own position relative to the landmark by a two-dimensional image. Since the coordinates and the landmark orientation are previously known, the robot also determines its own position in the room.

![Figure 1. An example of an artificial landmark indoor location.](image)

To conduct an experiment, a mobile robot that consists of two leading wheels, one support wheel, two motors with encoders mounted on the shaft, motor drivers, a video camera, control and power systems has been designed. The control system utilizing the described algorithm is realized on the basis of a controller and a computer.

The interaction of the control cards is carried out through a serial communication interface. The motor control function is implemented on the basis of a PID controller. The input data for its program are the coordinates \((x_i, y_i)\) of \(i\)-node of the destination trajectory received from the computer in real time.

Let us consider an algorithm of the motion control for a three-wheeled robot according to the coordinates stored in its memory with the use of the navigation system that includes a gyroscope and rotation sensors mounted on the shafts of the leading wheels (Figure 2).
Figure 2. Description of an algorithm for the robot motion control.

The starting location of the robot is position a.
The location of the robot is described by variables x, y, φ. The motion trajectory consists of a set of vertices (1-6) described by coordinates xi, yi connected by segments. Angle φ describing the rotation of the robot is calculated from the X-axis counter-clockwise to the X’ axis of the robot wheels (approximately -40º, position b).

Vector \( \vec{D} \) is the current direction coinciding with the Y’-axis of the local coordinate system of the robot. During the movement, the robot is directed to point 1 \((x_1; y_1)\). When the robot crosses a circle with a diameter of 1 m (shown as a dashed line), it is considered that the point is reached and the next target is point 2 \((x_2; y_2)\).

The motion occurs sequentially between the points until the last point is reached. To move to the next point, target vector \( \vec{T} \) and direction vector \( \vec{D} \) are calculated:

\[
\vec{T} = (x_2; y_2) - (x; y);
\]

\[
\vec{D} : Dx = -\sin \varphi; Dy = \cos \varphi.
\]

The cosine of the correction angle of the direction between \( \vec{T} \) and \( \vec{D} \) is calculated with the formula:

\[
\cos \alpha = \frac{-Tx \cdot \sin \varphi + Ty \cdot \cos \varphi}{\sqrt{Tx^2 + Ty^2}}.
\]

The correcting vector of direction (shown in red) \( \vec{C} = \vec{T} - \vec{D} \) is calculated. To transfer the vector to the local coordinate system of the robot, the clockwise rotation matrix is used [7]:

\[
\begin{bmatrix}
Cx’ \\
Cy’
\end{bmatrix} =
\begin{bmatrix}
\cos \varphi & \sin \varphi \\
-\sin \varphi & \cos \varphi
\end{bmatrix}
\begin{bmatrix}
Cx \\
Cy
\end{bmatrix}.
\]

If \( Cx > 0 \), the robot rotates clockwise to the right, otherwise it goes to the left. During the movement of the robot, x, y, φ are updated according to the data from the robot sensors, φ is determined with the help of a gyroscope. The speed of motion is calculated as follows:
\[ V = \frac{1}{2} (V_R + V_L), \]

where \( V_R \) and \( V_L \) are the linear velocities of the right and left wheels in m/sec respectively:

\[ \Delta x = V \cdot \cos \varphi \cdot \Delta t, \quad \Delta y = V \cdot \sin \varphi \cdot \Delta t. \]

At each step, these increments are added to the coordinates.

To determine the coordinates of an object in the space by its two-dimensional image, the expressions describing its projection onto the plane using a camera are needed.

Modern photo and video cameras are well described with the use of a mathematical model called a projective or pinhole camera [8, 9]:

\[ x = PX = K[R \mid t]X, \quad \text{where:} \]

- \( x \) – the coordinates vector of the point in the image \((x, y, 1)\);
- \( K \) – a 3x3 matrix of internal camera parameters;
- \( R \) – a 3x3 rotation matrix;
- \( t \) – a 3x1 transference vector;
- \( X \) – a world coordinate \((X, Y, Z, 1)\).

The matrix of camera \( K \) is described in the following form:

\[
K = \begin{bmatrix}
fx & 0 & cx \\
0 & fy & cy \\
0 & 0 & cy
end{bmatrix}.
\]

Transformation matrix \( P \) has a 3x4 size and is described as:

\[
P_{3\times4} = K[R \mid t] = \begin{bmatrix}
f \cdot kx & 0 & cx & tx \\
f \cdot ky & 0 & cy & ty \\
0 & 0 & 1 & tz
end{bmatrix}.
\]

where \( f \) – a focal length in pixels;
\( kx, ky \) - scale coefficients of the pixel size and distance. Ideally, if the camera pixels are square, these parameters are equal to 1. Real cameras are a little different from 1 on the axes.

\( K \) is called an internal parameters matrix. The matrices of external parameters \( R \) and \( t \) define the rotation and transfer of the camera in space. These matrices are calculated as a result of the three-dimensional reconstruction for each image and provide the calculation of the Cartesian coordinates of the camera.

Thus, if several points of an object and their real positions in space relative to each other are known, it is possible to build an equation system and determine the coordinates on its basis.

In Figure 1, the landmark boundary is a square with vertices: \((0,0,0)\); \((s, 0, 0)\); \((s, s, 0)\); \((0, s, 0)\), where \( s = 0.15 \) m is the landmark size equal to 0.15 m. To build and solve such equations in computer vision systems, the cross-platform library of computer vision algorithms OpenCV is used [10].
3. **Robot navigation algorithm**

Based on the approach described above, an algorithm is developed that consists of the following steps:

- **Step 1.** Information from a camera is processed using a computer. First, the image is searched for a geometric shape that can potentially be an artificial landmark (Figures 1, 2). The landmark has the form of a square. Because of the perspective effect, a square landmark in a two-dimensional image is usually observed in the form of a convex quadrangle. The image obtained from the digital camera on the robot is a bitmap, and it consists of an array of individual points or pixels. To search for geometric figures, the contours of objects in the image must be represented as a set of vectors. Then the contour analysis of the image is performed.

- **Step 2.** As a result of the contour analysis of the image obtained by the mobile robot movements, for example, around a production room, not only the required landmark boundaries will be detected. The contours of objects emerging from the environment, the outlines of doors, cabinets, shelves will enter the resulting set of vectors. If the image is not pre-converted, each small detail, each contour on the textures of different surfaces will be taken into account when searching for marks.

- **Step 3.** To reduce the number of contours, the image is converted into a gray scale. Then the threshold filter converting dark and light areas into white and black areas respectively is used. The objects, not highlighted enough against the background, will be merged with dark or white areas and excluded from further calculations. It should be noted that the correct value of the filtering threshold strongly depends on the illumination. Therefore, it is advisable to use more complex but less fast filters, for example, the adaptive Canny filter [11]. As a result, a list of contours is created, each of which consists of an array of vertices. Usually they are more numerous than the landmarks in the image. Therefore, the small contours that are not quadrangles are excluded from the further analysis.

- **Step 4.** The code of a landmark is determined, wherefore the image area bounded by each contour is analyzed. The use of a noise-protected code in the landmarks allows detecting contours that are not boundaries of the landmark. The image area bounded by the contour is copied to a separate memory area and is distorted to a square shape in order to exclude the perspective effect. By counting the colors of pixels in each element of the landmark, a matrix with a code consisting of 0 (black squares) and 1 (white squares) is formed. The matrix is converted into a number in decimal notation corresponding to the landmark number. Each row of a landmark encodes 2 bits from the high to the low, as shown in Table 2.

| Bits in binary notation | Bits in a noise-protected code |
|-------------------------|--------------------------------|
| 00                      | 1000                           |
| 01                      | 1011                           |
| 10                      | 0100                           |
| 11                      | 0110                           |

For example, the code on the left in Figure 2 looks like:

1 0 0 0 0 = 00
1 0 1 1 1 = 01
0 1 0 0 1 = 10
0 1 1 1 0 = 11
0 1 0 0 1 = 10

The encoded number in decimal notation: 00 01 10 11 10 = 110.

Code on Figure 1 – 00 11 10 11 00 = 236.
• Step 5. The coordinates of the camera from which the image was obtained are determined relatively to the found landmark with the use of the equation (1). The list of the landmarks with their identification codes, locations and orientations in space is compiled previously and is stored in the memory of the mobile robot (Figure 3). When a landmark with a specific code is detected, the robot using its known and calculated coordinates of the camera relative to the landmark, adjusts its current coordinates in the random access memory.

![Figure 3. The parameters of a landmark: a – an original image with calculation results on a computer display, b – an image after filtering.](image)

4. Conclusion
The presented method made it possible to detect landmarks with a size of 15 cm and their positioning in space with the use of a video camera with a resolution of 640x480 pixels at a distance of up to 6 m with a frequency of 15-20 frames per second. The system Arduino Mega 2560 is used as a motor controller. The computer vision system is realized on the basis of a single-board computer Raspberry Pi 3. The accuracy of the coordinates determination at a distance of 1 m from the landmark is ± 0.02 cm. Thus, the use of artificial landmarks in conjunction with the encoder system allows creating a relatively simple system of the mobile robot indoor navigation.

References
[1] Kolodziej K W., Hjelm J 2006 Local Positioning Systems: LBS Applications and Services (CRC Press)
[2] Huang Y and Cao Q 2010 Industrial Robot: An Int. J. 37(2) 193–201
[3] Zhong X, Zhou Y and Liu H 2017 Int. J. Adv. Robot Syst. 14(1) 1–13
[4] Kristensen S, Horstmann S, Klandt J, Lohner F and Stopp A 2001 Proc. of the 2001 IEEE International Conference on Robotics and Automation (Seoul Korea) 2590-2595
[5] Kadir T and Brady M 2001 Int. J. of Computer Vision 45(2) 83–105
[6] Chung H, Ojeda L and Borenstein J 2001 Accurate mobile robot dead-reckoning with a precision-calibrated fiber-optic gyro IEEE Trans. on Robotics and Automation vol 21(1)
[7] Balakrishna R. and Ghosal A 1995 Modeling of slip for wheeled mobile robot IEEE Transaction on Robotics and Automation vol 11(1) 126–132
[8] Zhang A 2000 Flexible New Technique for Camera Calibration IEEE Transactions on Pattern Analysis and Machine Intelligence vol 22(11) 1330-1334
[9] Fusiello A, Trucco E and Verri A 2000 Machine Vision and Applications 12(1) 16–22
[10] Kaehler A and Bradski G Learning OpenCV 3. Computer Vision in C++ with the OpenCV Library (O’Reilly Media) p 1024
[11] Chaithra N M and Ramana Reddy K V 2013 Int. J. of Eng. and Adv. Technology 2(6) 243-247