DOI: 10.18721/JCSTCS.13102
УДК 004.896

APPLICATION OF THE COMPUTER VISION SYSTEM FOR CONTROLLING A MOBILE ROBOT IN A DYNAMIC ENVIRONMENT

F. Daeef

Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russian Federation

The article proposes a new approach to the use of computer vision when controlling a robot in a dynamic environment. While moving along an unchanged path to the target point, a robot can encounter any new object (static or moving). We describe a visual analysis to determine the detection distance of moving objects to prevent collisions with the robot in a timely manner. An obstacle detection algorithm in the robot zone was developed based on data from an RGB-D video camera using computer vision methods. Based on ROS in a Gazebo virtual environment with a Turtlebot robot kit and an open source library (opencv), we adopted software implementation of the developed approaches which confirmed their applicability to the detection of objects in the mobile robot environment.

Keywords: mobile robots, dynamic environment, navigation, vision system, object detection.

Citation: Daeef F. Application of the computer vision system for controlling a mobile robot in a dynamic environment. Computing, Telecommunications and Control, 2020, Vol. 13, No. 1, Pp. 19-30. DOI: 10.18721/JCSTCS.13102

ПРИМЕНЕНИЕ СИСТЕМЫ ТЕХНИЧЕСКОГО ЗРЕНИЯ ПРИ УПРАВЛЕНИИ МОБИЛЬНЫМ РОБОТОМ В ДИНАМИЧЕСКОЙ СРЕДЕ

Ф. Даееф

Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

Предложен новый подход к применению системы технического зрения при управлении мобильным роботом в динамической среде. Во время движения по неизменной траектории к целевой точке на пути робота может появиться любой новый объект (статический или подвижный). В статье описан визуальный анализ для определения расстояния, на котором должно происходить обнаружение объектов, для своевременного предотвращения столкновения с роботом. Алгоритм обнаружения препятствий в зоне робота разработан на основе визуальных данных от RGB-D видеокамеры с помощью методов компьютерного зрения. На основе фреймворка ROS в виртуальной среде Gazebo, а также при помощи комплекта Turtlebot и библиотеки с открытым исходным кодом (opencv) написана программная реализация разработанных подходов, подтвердившая их применимость к обнаружению объектов в среде мобильного робота.

Ключевые слова: мобильный робот, динамическая среда, навигация, система технического зрения, обнаружение объектов.

Ссылка при цитировании: Даееф Ф. Применение системы технического зрения при управлении мобильным роботом в динамической среде // Информатика, телекоммуникации и управление. 2020. Т. 13. №1. С. 19-30. DOI: 10.18721/JCSTCS.13102

Introduction

Moving object detection in video streams is an interesting problem with a very wide range of vision-based application fields, such as action recognition [1], traffic control [2], industrial control [3], identification of human behavior [4] and intelligent video surveillance [5]. The general idea of detecting moving objects is to represent a set of related image pixels in a video sequence having a coherent motion in time (time aspect) and semantic similarity in the image space (spatial aspect) [6].

In this article we try to solve this problem for the purpose of robot navigation. Global path planning algorithm usually uses priori information to build a complete model of a structured environment, and then tries to find the best possible solution. But information is scarce in unknown or unstructured environments, so users need to combine the route planning method with local or reactive navigation using built-in sensors to locally observe small fragments of the environment at any time. A problem of detecting moving objects and responding to obstacles arises in this scenario. The most common approaches are: firstly, a proximity sensor belt (ultrasonic, infrared, ...) mounted on the vehicle, allowing for discrete scanning of the space around the robot; secondly, a rotating laser beam, often associated with a viewing system, which leads to a continuous assessment of the free area around the vehicle. The issue of accounting for moving obstacles in the management of a mobile robot still requires further research. We have a look at different ways to solve the problems presented in a number of studies. In the papers [7, 8] where laser scanners are used to detect obstacles the approaches described have limitations on the type of object's movement and the environment in which the robot can move; laser scanners are also rather expensive. In other studies, robot navigation is described using an on-board camera and video data. The purpose of detecting a moving object is to take a video sequence from a fixed / moving camera and output a binary mask representing moving objects for each frame of the sequence. However, this is not an easy task due to many problems and difficulties that arise when using a camera to capture a video sequence of moving objects. We categorize the existing methods to solve the problem in Table 1.

Table 1

Moving object detection from moving camera

Category	Positive points	Negative points
Background modeling [9-11]	• Moderately complex • Good for real time applications • Providing good object's silhouette	• Not good for free camera motion • Accuracy is highly dependent on background model
Trajectory classification [12, 13]	• Providing good object's trajectory over time • Moderately complex	• Very sensible to noise • Does not provide information on object's silhouette • Accuracy is highly dependent on motion tracker mode
Object tracking [14-16]	• Good performance with all camera motion • Moderately complex	• Does not provide information on object's silhouette • Needs initial good selection of the object
Low rank and sparse representation [17-19]	• Highly accurate • Providing good object's silhouette	• Requires a collection of frames • Not suitable for real-time applications • Highly complex

In this work, we try to solve these problems for navigation of a mobile robot in a dynamic environment on an unchanged path using the on-board vision system (an RGB-D Camera).

Problem statement

Let's have a transport robot on a flat underlying surface. There are static obstacles (walls, columns, tables, chairs, etc.), as well as moving obstacles (people, other robots), which are physical bodies that conform to the laws of dynamics. Let us also have a planned path for the robot, which cannot be changed during movement, in order to transfer material to the target point in the coordinates of the room. Such tasks arise during automation of transport operations in warehouses in which there are mobile objects –robots, people, etc. As a result, the system must position the robot at the target point, moving along the planned route and avoiding moving obstacles using high-speed adaptation method. It uses the computer vision system for this purpose while minimizing the movement time. Such transport robots, as a rule, can move at the maximum speed V_{rmax} = 1÷1.5 m/s and acceleration A_{rmax} = 0.1÷0.7 m/s^2. In our task we consider that the maximum speed and maximum acceleration of the robot V_{rmax} = 1.5 m/s, A_{rmax} = 0.3 m/s^2.

During the task of transportation, the robot should not deviate from the route, and has to avoid collisions with objects (Fig. 1), which can appear on the way. For this, we need to detect all objects (moving and static) on the way by using a vision system. In our task, we have no condition on the movement direction of objects, so objects can move in any direction. At the same time, they are solids with an unchanged shape, the maximum value of speed is V_{omax} = 1.5 m/s.

First of all, we have to calculate the distance for detecting the object, on which we have to slow down, because this is one of the most important factors for evaluating our work.

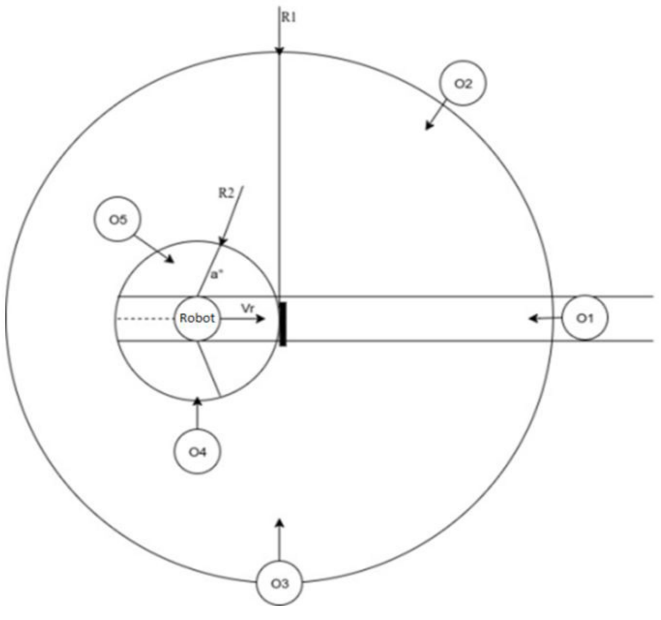

Fig. 1. Example of an environment in which a mobile robot moves, containing moving obstacle objects O1–O5, here R1 = S_n (the distance at which a moving obstacle is to be detected), R2 = S_r (distance to the robot full braking)

Suppose we have robot acceleration A_r, speed V_r, then the braking distance S_r is determined by the formula:

$$V_r = \sqrt{2*(-A_r)*S_r} \tag{1}$$

$$S_r = \frac{V_r^2}{2*(-A_r)}. \tag{2}$$

In our study the maximum velocity of the robot is V_{rmax} = 1.5 m/s, the maximum acceleration is $A_r = \pm$ 0.3 m/s^2, by the formula we get S_r = 3.75 m. In this paper, we consider a dynamic environment with moving objects. So, in our case, the formula for determining the distance S_n, for detecting obstacles moving at a constant speed, should be:

$$S_n = S_r + V_o * T_t ,$$

where V_o is the maximum speed of the moving object; T_t – time to stop, which can be calculated:

$$T_t = \frac{V_r}{-A_r} .$$

At the maximum acceleration and speed of the robot, the time to stop T_t = 1,5 / 0,3 = 5 s. Thus, the detection of objects should be at a distance not less than S_n = 3,75 + 1,5·5 = 11,25 m, in case the object moves towards the robot.

Methodology development

Suppose we have a point with coordinates [X_w, Y_w, Z_w] in the world coordinate system. We want to translate these coordinates into a robot camera coordinate system [X_c ,Y_c , Z_c] and into image coordinate system [X_{img} ,Y_{img}]. To do this we have to build matrix P, to convert the world 3D coordinates to 2D image coordinates. Since we are using a digital camera, we will use a "pinhole" – a camera with a small hole instead of a lens or with a lens that simulates this effect. So we can describe the projection matrix in this way:

$$P = K * [R|t] , \tag{3}$$

where K is the internal matrix of transformation of three-dimensional coordinates of the camera into two-dimensional coordinates of the image (Fig. 2); [$R|t$] – the extrinsic matrix transformation of the camera which describes the camera in the world coordinate system, and the direction of its view.

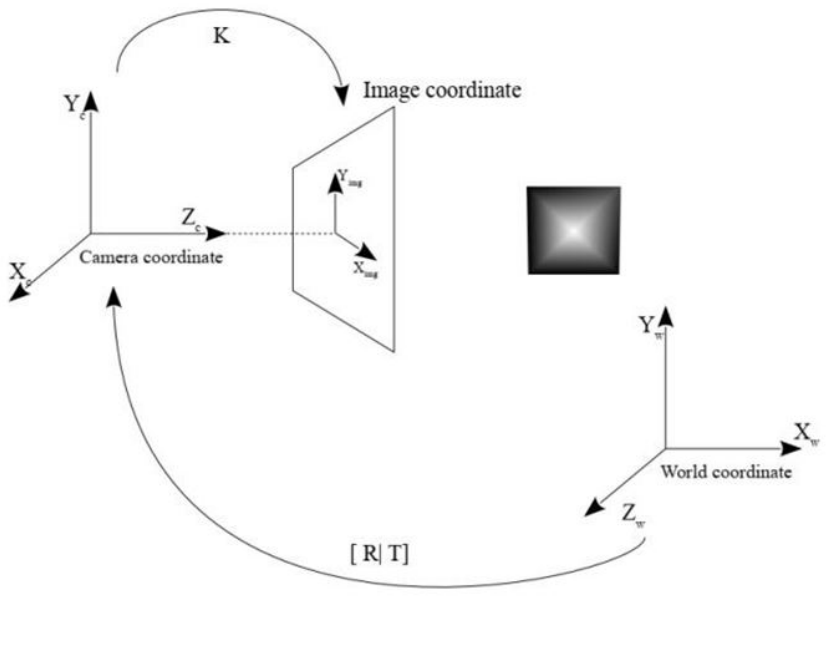

Fig. 2. Conversion from world coordinates to image coordinates

$$\begin{bmatrix} X_{img} \\ Y_{img} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{F}{Z_c} & 0 & \frac{w}{2} \\ 0 & \frac{F}{Z_c} & \frac{h}{2} \\ 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} X_c \\ Y_c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{F}{Z_c}(X_c) + \frac{w}{2} \\ \frac{F}{Z_c}(Y_c) + \frac{h}{2} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{F}{X_w + Z^{`}}\left( Z_w + X^{`} \right) + \frac{w}{2} \\ \frac{F}{X_w + Z^{`}}\left( Y_w + Y^{`} \right) + \frac{h}{2} \\ 1 \end{bmatrix}. \qquad (4)$$

Thus, we have a projection from the world coordinate system to the camera coordinate system; therefore, we can accept that $Z_c = X_w + Z`$ as the sum $X_w$, $Z`$ also for X_c, Y_c.

Since we assume we know the movement characteristics of the robot (position, speed and acceleration), so we will use the camera coordinate system [X_c, Y_c, Z_c].

Analysis of the object movement in the camera coordinates

We accept that objects move on a flat surface; therefore, we consider the movement of the objects in the XY plane in the robot coordinate system. We will try to describe how the real move of the objects in the robot coordinates system reflected on the image coordinates.

Depth axis movement (object speed on Z axis). To find how the movements of objects along the Z axis (depth axis) can be reflected in the image from the robot camera, we consider the relationship between the change in length of the line segment from object and the change in depth of the object. Let's consider two points on the image with coordinates (X_{1img}, Y_{1img}) (X_{2img}, Y_{2img}), the distance between these two points L_t:

$$\left(L_t \right)^2 = \left(X_{1timg} - X_{2timg} \right)^2 + \left(Y_{1timg} - Y_{2timg} \right)^2.$$

Using equation (4) we can rewrite as:

$$\left(L_t \right)^2 = \left(\frac{F * X_{1tc}}{Z_{1tc}} - \frac{F * X_{2tc}}{Z_{2tc}} \right)^2 + \left(\frac{F * Y_{1tc}}{Z_{1tc}} - \frac{F * Y_{2tc}}{Z_{2tc}} \right)^2.$$

Suppose that we observe two points belonging to one object, which does not change its shape or rotate around any of its axes, in other words the object is either static or moving in a straight line, so that $Z_{1c} = Z_{2c}$ in the camera coordinate system. Then we can rewrite:

$$\left(L_t \right)^2 = \left(\frac{F * \left(X_{1tc} - X_{2tc} \right)}{Z_{tc}} \right)^2 + \left(\frac{F * \left(Y_{1tc} - Y_{2tc} \right)}{Z_{tc}} \right)^2 = \frac{\left(F * Lx \right)^2 + \left(F * Ly \right)^2}{{Z_{tc}}^2}. \qquad (5)$$

Now, let's look at the change of the length of this line segment between two frames L_t, L_{t+1}

$$\frac{\left(L_t \right)^2}{\left(L_{t+1} \right)^2} = \frac{\dfrac{\left(F * Lx \right)^2 + \left(F * Ly \right)^2}{{Z_{tc}}^2}}{\dfrac{\left(F * Lx \right)^2 + \left(F * Ly \right)^2}{{Z_{t+c}}^2}} = \frac{{Z_{t+c}}^2}{{Z_{tc}}^2}. \qquad (6)$$

From this equation (6) we notice that $(Z)^{`} = \frac{-(L)^{`}}{L^2}$ which means that change in the length of a line segment from one object is inversely related to the change of the depth of the object. We can calculate a change of object coordinate in Z axis (depth information), but it is relative to depth value. Thus, we require information about depth (approximate) to calculate the change. These changes of depth information represent the speed of the object along Z axis. For this purpose, we use pixel depth data from the matrix of a digital RGB-D camera.

Vertical axis movement (object speed on X axis). To demonstrate the way object movement on the vertical axis is displayed on the image in the robot camera, consider one point with image coordinates X_{img}, Y_{img}.

Based on equation (4), we can write X_{img}, X_{img} as follows:

$$X_{img} = \frac{F}{Z_c}(X_c) + {}^{w}\!/_{2} \qquad => \qquad \frac{F}{Z_c}X_c = X_{img} - {}^{w}\!/_{2}.$$

Consider the change in its coordinates between frames:

$$\frac{\frac{F}{Z_{ct}}X_{ct}}{\frac{F}{Z_{ct+1}}X_{ct+1}} = \frac{X_{imgt} - {}^{w}\!/_{2}}{X_{imgt+1} - {}^{w}\!/_{2}},$$

$$\frac{X_{ct}}{X_{ct+1}} = \frac{X_{imgt} - {}^{w}\!/_{2}}{X_{imgt+1} - {}^{w}\!/_{2}} * \frac{Z_{ct}}{Z_{ct+1}}. \tag{7}$$

From this equation we can conclude that there are two factors affecting the movement of points in the image on axis X_{img}. The first factor is the movement of the point itself in space along *X* axis. The second one is its movement along the depth axis in such a way that both approximation and removal of the image of the controlled point eliminates the effect on the image of its movement along the depth axis.

The proposed approach

The general concept of our proposed method (detection of moving objects based on featured fragments) is shown in Fig. 3. As seen, the first step is the initialization of the featured fragments using the first set of frames. Next, we go in a cycle:

- Find comparisons of the featured fragments on the next frame.
- Then calculate the change in the position and length of each fragment between two frames.
- Update the fragment data.

After every round we send the fragment data to the robot control unit, where changes are analyzed and decisions are made for control of the robot speed. Color and depth frame sequences from an RGB-D camera comprise the inputs of the system. The goal is to provide information about the presence of moving or non-moving objects to the robot controller in order to decide (increase, decrease or not change) the speed of the robot.

Initialize featured fragments. In our task we propose that we don't have to find the whole moving object in the frame but the most interesting part of the object only for our task. We suppose this part is the edges that separate the body of the object from the background or other objects. That is a very widely studied problem called edge detection. To solve this kind of problem we use the Canny edge detector[1] developed in 1986 by John F. Canny. It uses a multi-stage algorithm to detect a wide range of edges in images. As a result of the Canny algorithm, we get a black and white picture where white pixels form the borders or edges that separate the objects. For better precision we have to filter the results of edge detection.

[1] https://en.wikipedia.org/wiki/Canny_edge_detector

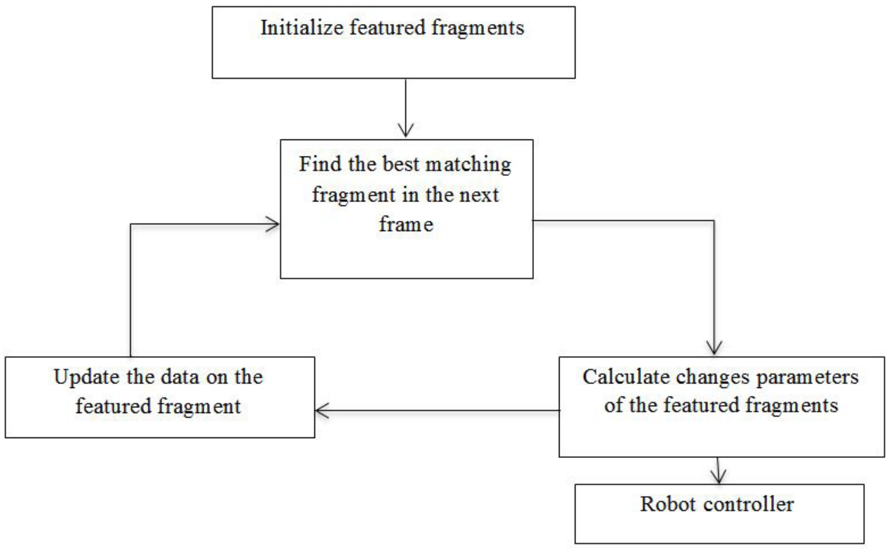
Initialize featured fragments
Find the best matching fragment in the next frame
Update the data on the featured fragment
Calculate changes parameters of the featured fragments
Robot controller

Fig. 3. Scheme of the proposed approach

Filtering featured fragments. The task of filtering is to remove false and unused points and fragments from image results after the edge detection phase. In order to apply such filtering, we need contours which represent curves connecting all the continuous points (along the boundary of an object) of the same color or intensity. Filtering is carried out on two bases:

- The length of the fragment itself (using fragments with length starting from 32 pixels).
- The depth of the point (up to 10 m).

Find the best matching fragment in the next frame. For the next step we propose to convert these fragments of the images into a set of segment lines and represent their by the two points ends segment. We use two methods for this step:

- The Hough transform to find imperfect instances of objects within a certain class of shapes (segment line) by a voting procedure.
- We use two ends of the contours that we found earlier as the two ends of a line segment.

As a result, we get a list of pairs of points we track between frames using an optical flow algorithm, based on the Lucas–Kanade method[1].

Calculating changes parameters of the featured fragments between two frames (determining the speed of objects)

We represent each pair of points (single segment line) as a separated object and try calculating the change in the length and position of this object (segment line) between two frames to find the speeds on vertical axis V_{ox} and depth V_{oz}.

1. Speed V_{oz}

We use an RGB-D camera that gives us depth data. For our task it is very important to measure depth error and how it increases with depth value. Khoshelham [2] studied a model for measuring depth error in the Kinect model camera (RGB-D) and concluded that the uncertainty of depth measurement is proportional to the square of the depth value $\sigma_z = \frac{1}{f}\sigma_d d^2$, where d is depth σ_d standard deviation, f is the focal length. For Intel® RealSense™ Depth Camera D435i errors increase quadratically from a few millimeters at a distance of 0.5 m to ~5 cm at the maximum distance of 10 m.

Therefore, we cannot get depth changes between frames from the depth data of the camera, but we can use the depth data from equation (6) to find the speed of an object:

[1] https://en.wikipedia.org/wiki/Lucas%E2%80%93Kanade_method

$$\frac{(L_t)^2}{(L_{t+1})^2} = \frac{Z_{t+c}{}^2}{Z_{tc}{}^2}, \tag{8}$$

where L_t is the length of a segment line at frame t and Z_{tc} is the depth of the object from the robot at the frame t.

2. Speed V_{ox}

We can calculate the projection of the velocity of object V_{ox} relative to the coordinate system of the mobile robot by the formula:

$$\Delta s_{yr} = \frac{X_{ct}}{X_{ct+1}} = \frac{X_{imgt} - w/2}{X_{imgt+1} - w/2} * \frac{Z_{ct}}{Z_{ct+1}}$$

$$V_{ox} = \Delta s_{yr} / t. \tag{9}$$

Testing the developed approaches

To test our method, we performed a precision-recall curve. Using virtual implementation in an environment (ROS), we examined two metrics (precision-recall) for detecting moving objects at the pixel level and at the object level. The results are shown in Fig. 4.

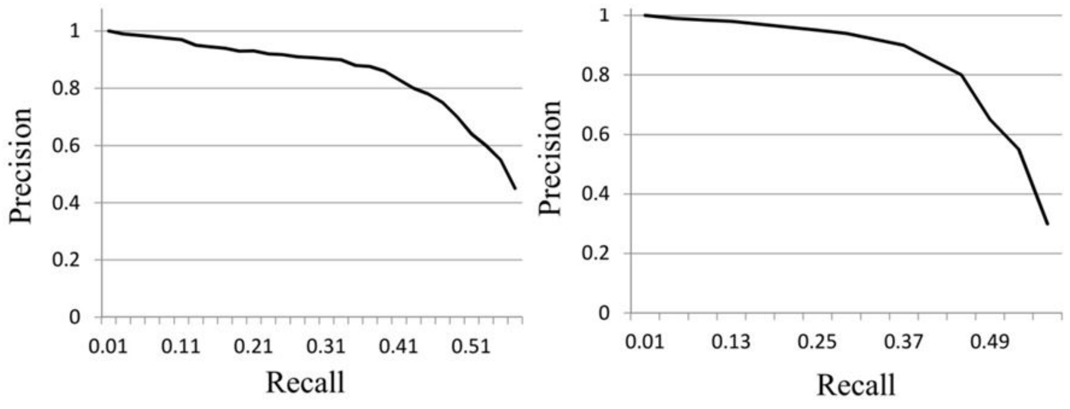

Fig. 4. Quantitative analysis of our approach
Precision-recall curve at the pixel level (left) and at the object level (right)

We also compared our results to a study conducted with the use of lidar [20]. Table 2 shows our result do not deviate much from lidar results, although we had only an RGB-D camera instead of some very expensive pieces of equipment.

Table 2

Comparison of the proposed method to lidar detection

	#Grd TruthObj	#Correct Obj	#Wrong Obj	Precision	Object Rcl	Sec./frame
Lidar [20]	52	48	0	1	0.9231	0.26
Proposed method	50	40	2	0.96	0.8	0.3

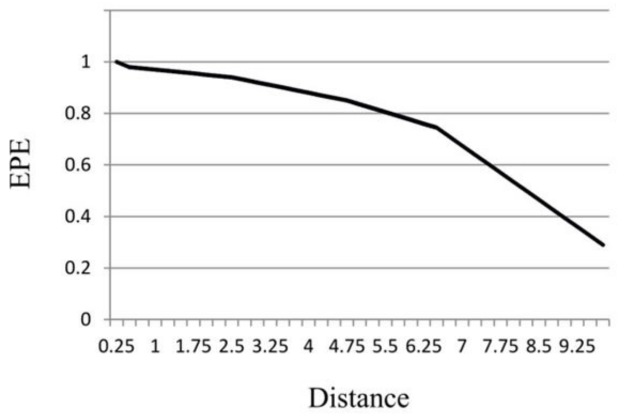

Fig. 5. Endpoint error (EPE) and the distance of the object from the robot

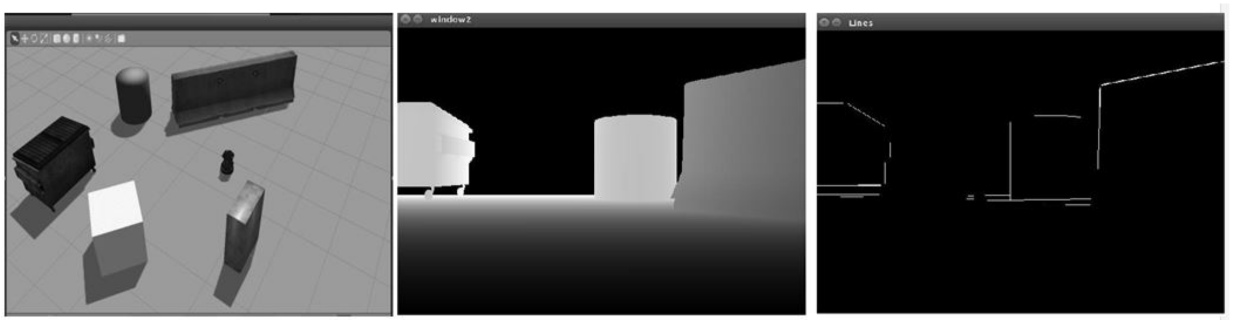

Fig. 6. Example of application based on ROS in Gazebo virtual environment with a Turtlebot robot kit. There are a general view of the scene (left), 3D image (center), and the result of the proposed method (right)

To analyze the tracking method, we consider a metric (EPE) that determines the error of the end-to-end point and how the distance of the object from the robot affects it. The end-point error is calculated by comparing the calculated optical flow vector V_{est} with the true optical flow vector V_{gt} for each point. We calculated this metric for each point in the list that we received in the previous step, in a static scene, because we know how they should move in the image (surface truth). Endpoint error is defined as the Euclidean distance between the two points: $A = | V_{est} - V_{gt} |$.

Using a virtual implementation, we were able to calculate the optical flow error EPE as a function of the distance of the object from the robot. We examined the values of the optical flow of static objects, the coordinates of which we know in the absolute system. We can calculate their coordinates on the images in each frame, then we calculate the error of the optical flow. Note in Fig. 5 that the closer is the object to the robot, the greater the error.

Conclusions

A detailed analysis of the environment of the mobile robot allowed us to solve the problem of calculating the braking distance and the distance of detection of obstacles to prevent collisions with them. The developed software implementation of the proposed approaches uses ROS (Robot Operating System) middleware (Fig. 6), which provides developers with libraries and tools for creating robotic applications. Our study conducted on a mobile robot with a caterpillar chassis confirms the applicability of our approach to analysis of a dynamic environment in real conditions when navigating transport robots in warehouse and workshop premises.

REFERENCES

1. **Wu S., Oreifej O., Shah M.** Action recognition in videos acquired by a moving camera using motion decomposition of Lagrangian particle trajectories. *IEEE International Conference on Computer Vision*, Nov. 2011, Pp. 1419–1426.

2. **Cucchiara R., Grana C., Piccardi M., Prati A.** Statistic and knowledge-based moving object detection in traffic scenes. *IEEE Intell. Transp. Syst.*, 2000, Pp. 27–32.

3. **Malamas E.N., Petrakis E.G., Zervakis M., Petit L., Legat J.D.** A survey on industrial vision systems, applications and tools. *Image Vis. Comput.*, 2003, Vol. 21 (2), Pp. 171–188.

4. **Hu W., Tan T., Wang L., Maybank S.** A survey on visual surveillance of object motion and behaviors. *IEEE Trans. Syst Man Cybern.*, 2004, Vol. 34 (3), Pp. 334–352.

5. **Kim J.S., Yeom D.H., Joo Y.H.** Fast and robust algorithm of tracking multiple moving objects for intelligent video surveillance systems. *IEEE Trans. Consum. Electron.*, 2011, Vol. 57 (3), Pp. 1165–1170.

6. **Yilmaz A., Javed O., Shah M.** Object tracking: A survey. *ACM Comput. Surv.*, 2006, Vol. 38 (4), P. 13.

7. **Khelloufi A., Achour N., Passama R., Che-rubini A.** Tentacle-based moving obstacle avoidance for omnidirectional robots with visibility constraints. *30th IEEE/RSJ Internat. Conf. on Intelligent Robots and Systems.*, 2017.

8. **Gerasimov V.N.** The motion control system of the mobile robot in environment with dynamic obstacles. *St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunication and Control Systems*, 2013, Vol. 5 (181), Pp. 100–108. (rus)

9. **Minaeian S., Liu J., Son Y.J.** Effective and efficient detection of moving targets from a UAV's camera. *IEEE Trans. Intell. Transp. Syst.*, 2018, Vol. 19 (2), Pp. 497–506.

10. **Wu Y., He X., Nguyen T.Q.** Moving object detection with a freely moving camera via background motion subtraction. *IEEE Trans. Circuits Syst. Video Technol.*, 2017, Vol. 27 (2), Pp. 236–248.

11. **Gong L., Yu M., Gordon T.** Online codebook modelling based background subtraction with a moving camera. *3rd International Conference on Frontiers of Signal Processing*, ICFSP, 2017, Pp. 136–140.

12. **Singh S., Arora C., Jawahar C.V.** Trajectory aligned features for first person action recognition. *Pattern Recognit.*, 2017, No. 62, Pp. 45–55.

13. **Yin X., Wang B., Li W., Liu Y., Zhang M.** Background subtraction for moving camera based on trajectory-controlled segmentation and label inference, KSIITrans. *Internet Inf. Syst.*, 2015, Vol. 9 (10), Pp. 4092–4107.

14. **Chen J., Sheng H., Zhang Y., Xiong Z.** Enhancing detection model for multiple hypothesis tracking. *Conference on Computer Vision and Pattern Recognition Workshops*, 2017, Pp. 2143–2152.

15. **Bagherzadeh M.A., Yazdi M.** Regularized least-square object tracking based on ℓ2,1 minimization. *3rd RSI International Conference on Robotics and Mechatronics*, ICROM, Oct. 2015, Pp. 535–539.

16. **Bouwmans T., Silva C., Marghes C., Zitouni M., Bhaskar H., Frelicot C.** On the role and the importance of features for background modeling and foreground detection. *Comput. Sci. Rev.*, 2018, No. 28, Pp. 26–91.

17. **Chau G., Rodriguez O.** Panning and jitter invariant incremental principal component pursuit for video background modelling. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, Pp. 1844–1852.

18. **Chen C., Li S., Qin H., Hao A.** Robust salient motion detection in non-stationary videos via novel integrated strategies of spatio-temporal coherency clues and low-rank analysis. *Pattern Recognit.*, 2016, No. 52, Pp. 410–432.

19. **Thomaz L., Jardim E., da Silva A., da Silva E., Netto S., Krim H.** Anomaly detection in moving-camera video sequences using principal subspace analysis. *IEEE Trans. Circuits Syst. I. Regul. Pap.*, 2018, Vol. 65 (3), Pp. 1003–1015.

20. **Ma Y., Anderson J., Crouch S., Shan J.** Moving object detection and tracking with doppler LiDAR. *Remote Sens*, 2019, No. 11, P. 1154.

Received 27.01.2020.

СПИСОК ЛИТЕРАТУРЫ

1. **Wu S., Oreifej O., Shah M.** Action recognition in videos acquired by a moving camera using motion decomposition of Lagrangian particle trajectories // IEEE Internat. Conf. on Computer Vision. Nov. 2011. Pp. 1419–1426.

2. **Cucchiara R., Grana C., Piccardi M., Prati A.** Statistic and knowledge-based moving object detection in traffic scenes // IEEE Intell. Transp. Syst. 2000. Pp. 27–32.

3. **Malamas E.N., Petrakis E.G., Zervakis M., Petit L., Legat J.D.** A survey on industrial vision systems, applications and tools // Image Vis. Comput. 2003. Vol. 21 (2). Pp. 171–188.

4. **Hu W., Tan T., Wang L., Maybank S.** A survey on visual surveillance of object motion and behaviors // IEEE Trans. Syst Man Cybern. 2004. Vol. 34 (3). Pp. 334–352.

5. **Kim J.S., Yeom D.H., Joo Y.H.** Fast and robust algorithm of tracking multiple moving objects for intelligent video surveillance systems // IEEE Trans. Consum. Electron. 2011. Vol. 57 (3). Pp. 1165–1170.

6. **Yilmaz A., Javed O., Shah M.** Object tracking: A survey // ACM Comput. Surv. 2006. Vol. 38 (4). P. 13.

7. **Khelloufi A., Achour N., Passama R., Che-rubini A.** Tentacle-based moving obstacle avoidance for omnidirectional robots with visibility constraints // 30th IEEE/RSJ Internat. Conf. on Intelligent Robots and Systems. 2017.

8. **Герасимов В.Н.** Система управления движением мобильного робота в среде с динамическими препятствиями // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. 2013. № 5 (181). С. 100–108.

9. **Minaeian S., Liu J., Son Y.J.** Effective and efficient detection of moving targets from a UAV's camera // IEEE Trans. Intell. Transp. Syst. 2018. Vol. 19 (2). Pp. 497–506.

10. **Wu Y., He X., Nguyen T.Q.** Moving object detection with a freely moving camera via background motion subtraction // IEEE Trans. Circuits Syst. Video Technol. 2017. Vol. 27 (2). Pp. 236–248.

11. **Gong L., Yu M., Gordon T.** Online codebook modelling based background subtraction with a moving camera // 3rd Internat. Conf. on Frontiers of Signal Processing. 2017. Pp. 136–140.

12. **Singh S., Arora C., Jawahar C.V.** Trajectory aligned features for first person action recognition // Pattern Recognit. 2017. No. 62. Pp. 45–55.

13. **Yin X., Wang B., Li W., Liu Y., Zhang M.** Background subtraction for moving camera based on trajectory-controlled segmentation and label inference, KSIITrans // Internet Inf. Syst. 2015. Vol. 9 (10). Pp. 4092–4107.

14. **Chen J., Sheng H., Zhang Y., Xiong Z.** Enhancing detection model for multiple hypothesis tracking // Conf. on Computer Vision and Pattern Recognition Workshops. 2017. Pp. 2143–2152.

15. **Bagherzadeh M.A., Yazdi M.** Regularized least-square object tracking based on ℓ2,1 minimization // 3rd RSI Internat. Conf. on Robotics and Mechatronics, ICROM. Oct. 2015. Pp. 535–539.

16. **Bouwmans T., Silva C., Marghes C., Zitouni M., Bhaskar H., Frelicot C.** On the role and the importance of features for background modeling and foreground detection // Comput. Sci. Rev. 2018. No. 28. Pp. 26–91.

17. **Chau G., Rodriguez O.** Panning and jitter invariant incremental principal component pursuit for video background modelling // Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. 2017. Pp. 1844–1852.

18. **Chen C., Li S., Qin H., Hao A.** Robust salient motion detection in non-stationary videos via novel integrated strategies of spatio-temporal coherency clues and low-rank analysis // Pattern Recognit. 2016. No. 52. Pp. 410–432.

19. **Thomaz L., Jardim E., da Silva A., da Silva E., Netto S., Krim H.** Anomaly detection in moving-camera video sequences using principal subspace analysis // IEEE Trans. Circuits Syst. I. Regul. Pap. 2018. Vol. 65 (3). Pp. 1003–1015.

20. **Ma Y., Anderson J., Crouch S., Shan J.** Moving object detection and tracking with doppler LiDAR // Remote Sens. 2019. No. 11. P. 1154.

Статья поступила в редакцию 27.01.2020.

THE AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

DAEEF Feras
ДАЕЕФ Ферас
E-mail: ferasit87@gmail.com