Software system for simultaneous localization and mapping based on video content analysis

Ivan Perić, Miroslav Kondić, Stefan Anđelić, Marko Jocić, Đorđe Obradović
Faculty of Technical Sciences/Department of computing and control engineering, Novi Sad, Serbia
ivanperic@uns.ac.rs, kondicm@uns.ac.rs, stefan.andjelic@uns.ac.rs, jocicmarko@gmail.com, obrad@uns.ac.rs

***Abstract*— This paper consists of the formal specification and implementation of the system for simultaneous localization and mapping based on video content analysis using one video camera, without usage of any additional sensors. Lack of sensors lowers the system price, but software complexity becomes very high. Shi-Thomasi corner detection method is used in system implementation, as well as Lucas-Kanade optical flow method, RANSAC algorithm for optical flow vector filtering and Kalman filter for correction of the camera position.**

I. INTRODUCTION

This paper holds the formal specification of the system for simultaneous localization and mapping based of video content analysis. Simultaneous localization and mapping (a.k.a. SLAM) is problem usually found in robotics. The basic idea lies in a fact that a moving robot should be able to map the space around him and to track its location within that space. The system presented in this paper uses only one video camera for that purpose, without additional sensors. System is implemented using computer vision techniques, numerical algorithms and others. The formal specification of the system will be presented in the next section of this paper. Third section contains system verification from the performance and accuracy viewpoints, while the fourth section contains some further research plans.

II. SYSTEM SPECIFICATION

Software system presented in this paper is divided into seven subsystems where each of them performs a group of similar tasks. By doing so, it's easier to follow the simultaneous localization and mapping process. Figure 1 contains the conceptual diagram of the system.

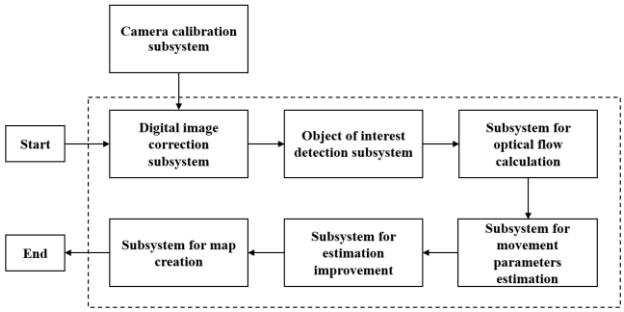

Figure 1. Conceptual diagram of the system

All subsystems will be presented in the next segments, by presenting theoretical concepts which are used in subsystems. An input to the whole system is a video recording, or the live camera feed. Every frame of the video recording goes through subsystems of the software system. Subsystems will process and modify them, before extracting the data which will be used in localization and mapping process.

2.1. Camera calibration subsystem

Camera lens characteristics and imperfections will be presented in this segment. Some of those imperfections can affect video processing in SLAM problem, and that's why the physics of the camera lens will be explained in the Figure 2 [1].

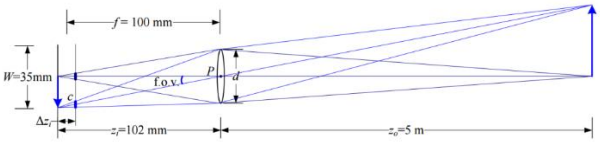

Figure 2. Camera lens model

The fact that video camera lens exists and that is not perfectly thin, results in image defects which are known as lens aberrations. The reason why spherical distortion of the image occurs (spherical aberration) is that the focus point of the light rays depends on the distance of the source and the optical axis. The consequence is that straight lines on the scene will be curved on the image. Spherical distortion removal is the problem that can be solved by camera calibration process. Types of image distortion are displayed in the Figure 3.

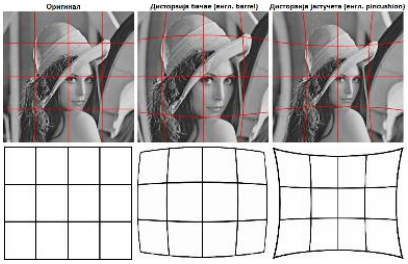

Figure 3. Image distortion: a) original image, b) barrel distortion, c) pincushion distortion

Camera calibration subsystem calculates camera lens parameters, which are then stored inside the system. Calibration process is based on recording of the pattern which has known proportions and characteristics from many angles and mapping projected points onto the points in 3D space. Every image with detected pattern defines a new equation which uses extrinsic and intrinsic parameters and distortion coefficients [2].

2.2. Digital image correction subsystem

The input of this subsystem is the input of the entire software system and each frame of the video content that is used for mapping and localization is going through this

subsystem. Every frame that is processed in this subsystem is forwarded to other subsystems.

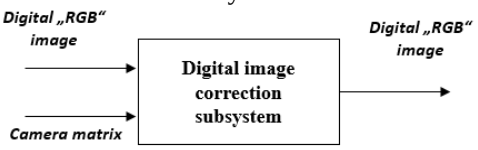

Figure 4. Digital image correction subsystem

Besides the digital image the correction is applied to, the output of the camera calibration subsystem is also brought as an input to this subsystem (Figure 4). Subsystem for the correction of digital images fetches those parameters, and corrects image distortion that is created as a result of the imperfections of the camera lens. The corrected digital image is used in the rest of the software system.

2.3. Objects of interest detection subsystem

To enable the analysis of motion in the video, it is necessary to define some key points which will be used to calculate motion parameters. This process can be divided into several phases. A block diagram of this subsystem is shown in Figure 5.

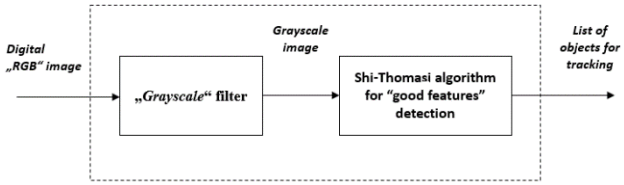

Figure 5. Objects of interest detection subsystem

Most of algorithms for the detection of good objects for tracking use the assumption that the input image is a function of the light intensity of the pixel depending on its coordinates. That is why a digital image is usually transformed into the image where every pixel represents the amount of brightness to the appropriate coordinates of the input image. For this purpose, grayscale filter is used. This approach eliminates complex analysis in the color space, and uses only 256 shades of gray.

Jianbo Shi and Carlo Thomas in their research [3] propose an algorithm for the detection of good tracking features. They show how to monitor the quality of the tracked objects in the process of motion detection by using dissimilarity measure, which quantifies the change in appearance between the first and the current frame of video content. The measure represents the diversity of the rest of the object which was detected in the first frame and its appearance in the current frame. When dissimilarity measure becomes too high, tracked object is not suitable for tracking anymore and needs to be abandoned and it is desirable to find another object. Shi-Thomasi detector represents the evolution of the algorithms presented by the Moravec [4] and Harris-Stephens in [5].

The method is based on the calculation of the autocorrelation matrix which describes the distribution of the gradient in a local neighborhood of a point. In fact, the difference in the intensity of displacement *(u, v)* is found in all directions:

$$E(u,v) = \sum_{x,y} w(x,y)\ [I(x+u, y+v)\ -\ I(x,y)]^2 \qquad (1)$$

The function *w(x, y)* represents the so-called, window function. It is usually rectangular or Gaussian window in which the pixels beneath have weights attached. *I(x+u,y+v)* represents the shifted intensity, and *I(x,y)* is the unmoved intensity. The idea of the whole process is to maximize the function *E (u, v)* for edge detection.

Of all the potential candidates, system chooses the *N* best. The selection process begins with all available candidates sorted by how good they are. Then the system selects the best of them, and eliminates all the other candidates who are close to him. The process continues until a top N candidates are selected.

2.4. Subsystem for optical flow calculation

This chapter contains a specification of the subsystem for optical flow calculation for objects in the scene. The main task of this subsystem is to determine displacement vector of the camera. In order to analyze the displacement of the camera or the scene, it is necessary to have at least two reference positions that will be used in order to calculate those displacement vectors. The whole system is based on the analysis of video content, so the consecutive frames of the video recording will be analyzed. A conceptual diagram of this subsystem is shown in the Figure 6.

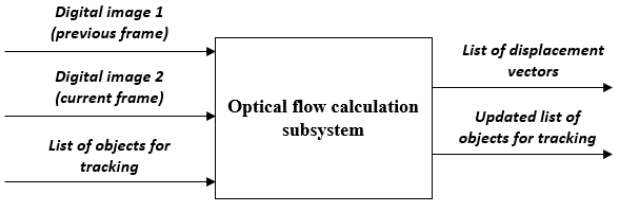

Figure 6. Optical flow calculation subsystem

Optical flow represents a pattern of displacement of objects, surfaces and edges on the scene. It is caused by a change of the relative position between the observer (eye or a camera) and the scene. It basically represents a distribution of the velocity vectors of tracked points. An example of the optical flow is shown in Figure 7.

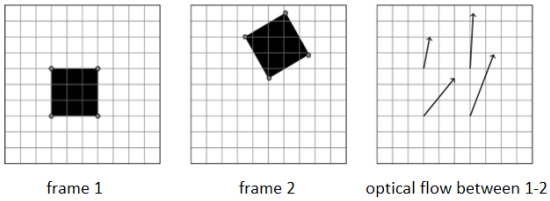

Figure 7. An example of the optical flow between two frames

In order to calculate the optical flow estimate in two dimensions, an assumption that $I(x,y,t)$ is a central pixel in the square n×n and that it is displaced by δ_x, δ_y in time δ_t to the position $(x+\delta_x, y+\delta_y, t+\delta_t)$ is introduced. Since the *I(x,y,t)* and $I(x+\delta_x, y+\delta_y, t+\delta_t)$ are the images with the same points, we get:

$$I(x,y,t) = I(x+\delta_x, y+\delta_y, t+\delta_t) \qquad (2)$$

The previous equation is known as the equation of the optical flow in the plane. Values of the change δ_x, δ_y i δ_t are not high. After this step, an approximation using Tailor's series is performed on the previous equation. The derivation process is explained in [6]. The result of the derivation process is an equation with two variables that cannot be solved. Then, the Lucas-Kanade method approximation is used as a solution to this problem. The basic idea of this method lies in the assumption that the optical flow field is spatially sustainable, and that all pixels that fall into a window of dimensions n × n have the same value of the velocity *v*, and that's why the optical flow in that window is constant. [7] The result is a set of equations that has more equations than the number of variables (Figure 8b). [8]

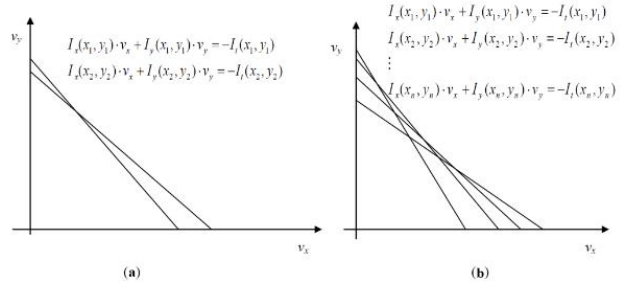

Figure 8. Intersection of a) two equations of the optical flow, b) multiple equations of the optical flow

In this case, the least squares method is used to approximate the unique solution of the system of equations. After the process of derivation, which is shown in [7], an expression for optical flow vector calculation is given (3):

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(p_n)^2 & \sum_i I_x(p_n)I_y(p_n) \\ \sum_i I_x(p_n)I_y(p_n) & \sum_i I_y(p_n)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(p_n)I_t(p_n) \\ -\sum_i I_y(p_n)I_t(p_n) \end{bmatrix}$$

The output of this subsystem is the set of all optical flow vectors for all tracked objects.

2.5. Subsystem for movement parameters estimation

Subsystem for movement parameters estimation has the task to calculate the intensity and angle of displacement of the camera, by using the analysis of the optical flow field for objects in the scene. Figure 9 shows a conceptual diagram of the subsystem for estimation of motion parameters.

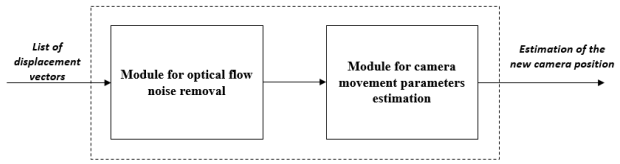

Figure 9. Subsystem for movement parameters estimation

Optical flow field can be quite noisy, especially if the scene is almost homogeneous, resulting in harder object detection and tracking. An example of the noisy optical flow field is given in Figure 10.

Figure 10. Noisy optical flow field

The basic idea behind the removal of the noise lies in the fact that most of the vectors have a similar direction. In this case, we can determine what is noise and what is not. For this purpose we used RANSAC algorithm, which, unlike the method of least squares regression does not have to include all data. It finds a subset that will have the best approximation, while other elements are classified as outliers, as shown in Figure 11.

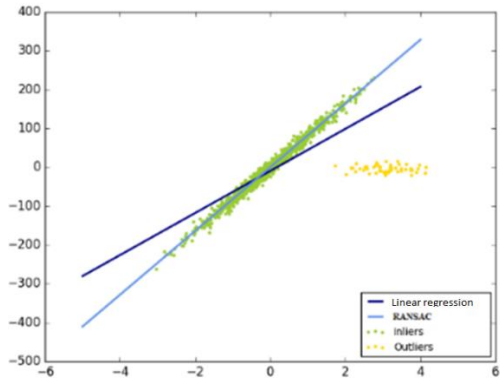

Figure 11. Difference between the least squares linear regression approach and RANSAC algorithm

After the filtering process, the resulting motion vector is calculated by simple averaging of remaining vectors. Then the new camera position is calculated, taking into account its orientation and the difference between the previous and the current angle.

2.6. Subsystem for estimation improvement

If the estimated value of the new camera position is used in the rest of the system in its original form, with no improvement, it may happen that the map of the movement looks unnatural. The supporting structure to which the camera is attached, for example a vehicle, won't usually be perfectly still. Small vibrations of the support structure will result in a trajectory that will not be perfectly straight, even if the camera moved by the straight line. If the camera is attached to the vehicle, vibration is logical phenomenon. When a man wears a camera, vibration will occur during the walk, or if the camera is held in hands the small shake of hands will result in a vibration. No matter how small these vibrations are, they will be detected at the transition between adjacent frames and they will bring noise to the system. The basic idea behind the methodology for improving the estimation lies in the fact that the camera moves according to the laws of physics, because the camera will generally be attached to the moving vehicle. It is not possible that a vehicle at high speed makes a huge

turning angle, for example 90 °. Those "sharp" angles will happen when the vibration affects the camera and can be eliminated if we take into account the laws of motion in physics. Based on the theory of Kalman filter [9], it can be concluded that precisely this filter could be applied to this kind of estimation improvement, even though there are many others.

An input of this subsystem consists solely of the coordinates of positions of the camera in the real world. Those coordinates are used for the prediction of the future position. Kalman filter uses the current value and a previous measurement, so the input of the subsystem will be the estimation of the current position of the camera and the previous position of the camera (Figure 12).

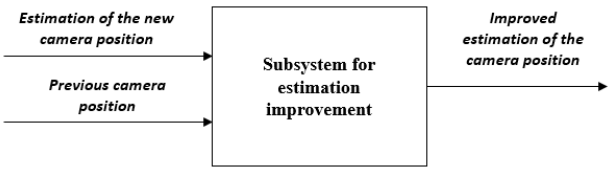

Figure 12. Subsystem for estimation improvement

As the model is formed in the discrete domain, the movement of the object is modeled by approximating the velocity and the acceleration by the difference equation I, wherein the time interval T equals time step, or T = 1.

$$v[n] = \frac{x[n] - x[n-1]}{T} \quad (4)$$

$$a[n] = \frac{v[n] - v[n-1]}{T} = \frac{x[n] - 2x[n-1] + x[n-2]}{T} \quad (5)$$

If equations above are included in the approximation of Newton's laws of motion in a discrete domain, the following expression is given:

$$x[n+1] = x[n] + v[n] \cdot T + a[n] \cdot \frac{T^2}{2} \quad (6)$$

Based on these equations, the model of the Kalman filter used in the system is formed.

2.7. Subsystem for movement map creation

The task of this subsystem is the mapping of the camera odometry in the corresponding metric system. An input of this subsystem is just the new camera position, which will be added to the map, as shown on Figure 13.

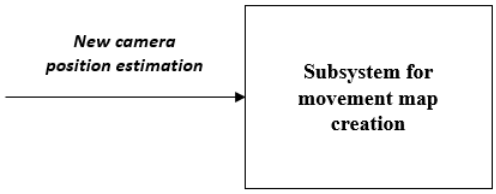

Figure 13. Subsystem for movement map creation

This system measures the distance in meters, so that this subsystem is able to switch metrics from pixels to meters before the drawing of the path is performed. To make something like this possible, several assumptions were introduced. The camera is fixed at a height *h* which is not changeable. This way, the width and height of the field of view that the camera covers can be measured, and this enables easy calculation of the length of any line in that plane. The width and height in pixels corresponds to a resolution in which the video content is recorded, and the width and height in meters is experimentally determined by measuring, after the camera is fixed to a suitable height *h*.

III. SYSTEM VERIFICATION

Verification of the accuracy of the system is done by empirical methods by analyzing plotted trajectory, while the performance verification is done experimentally. System performance was analyzed on machines with a single-core, and quad-code processors of the latest generation. For high-definition video, system reached 12 and 9 processed frames per second, depending on the machine. For video format with a resolution of 640x480, processing of 28 and 26 frames per second is achieved, which is enough to operate in real time. The accuracy of the system is satisfactory if the trajectory of the camera is longer. When turning the supporting structure of the system in a small area, a loss of short rotation vectors after RANSAC filtering happen and the resultant angle is not good (camera rotation around one of its corners) - Figure 14a. If steering relies only on the translational movement, without rotation of the camera, results are good - Figure 14b.

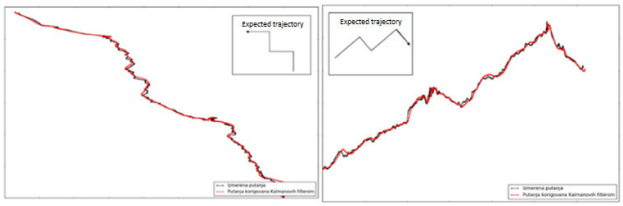

Figure 14. Resulting and expected trajectories in the case of the rotation around camera edge and translationar movement

IV. CONCLUSION

Based on the system verification, empirical methods show that it is operating well in a partially controlled conditions. If the camera rotation angle is too sharp on a small moving distance, there is a loss of some rotation vectors and the noise appears in the resultant trajectory. In order to avoid such a case, the camera may not be rotated around one of its corners. Steering should be performed by the translation of the supporting structure, for example vehicle, not by direct camera rotation. Further directions of development would involve the introduction of cheap sensors, which could follow the direction of the camera without relying solely on the analysis of the optical flow, and the camera wouldn't have to be fixed at a certain height in order to measure the length of distance traveled. The system would be more accurate because it would have two types of measurements - the measurements from the sensors and optical flow measurements. At the same time, system should be working faster because computationally demanding analysis of the optical flow could be simplified, but the cost of the system would be higher. In the approach used in this paper, system has a higher software complexity but it's cheaper and doesn't require any additional equipment besides one video camera and a computer.

REFERENCES

[1] F. Zhang, G. Yang and D. J. Bakos, Lucas-Kanade Optical Flow Estimation on the TI C66x Digital Signal Processor, Department of Computer Science and Engineering, University of South Carolina.

[2] G. Welch and G. Bishop, An Introduction to the Kalman Filter, Department of Computer Scrince, University of North Carolina at Chapel Hill, Chapel Hill, 2006.

[3] J. Shi and C. Tomasi, "Good Features to Track," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. CVPR94, p. 8, 1994.

[4] B. D. Lucas, Image Matching by the Method of Differences, Carnegie Mellon University, 1984.

[5] E. Hecht, 4th Edition, Optics, Addison-Wesley ISBN 9780805385663, 2002.

[6] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference, pages 147-151*, 1988.

[7] P. Corke, "Robotics, Vision and Control - Fundamental Algorithms in MATLAB," *volume 73 of Springer Tracts in Advanced Robotics*, 2011.

[8] J. L. Barron and N. A. Thacker, Computing 2D and 3D Optical Flow, Manchester: University of Manchester, 2005.

[9] "Camera calibration tutorial," OpenCV documentation, [Online]. Available: http://docs.opencv.org/master/d4/d94/tutorial_camera_calibration.html.