- [5] Bailey Bill. Connecting Simulink models to REST APIs. URL: <https://billbailey.io/2020/07/10/connecting-simulink-models-to-rest-apis/>.
- [6] Cirka L', Kal'uz M., Fikar M. On-line remote control of MATLAB simulations based on asynchronous communication model. *Matlab conference in Institute of Chemical Technology, Prague, 2013*. URL: http://dsp.vscht.cz/konference_matlab/MATLAB13/prispevky/014_cirka.pdf
- [7] Matsim: A sleek, intuitive interface for building Simulink models from a Matlab script. URL: <https://github.com/gave92/Matsim>.
- [8] Brennan Stephan. Write a Shell in C. URL: <https://brennan.io/2015/01/16/write-a-shell-in-c/>
- [9] Стопакевич А. А. Многомерная модель брагоректификационной установки косвенного действия. *Наукові праці ОНАХТ*, 2010. Т. 1., №38. С. 360-365.
- [10] Стопакевич А. А. Разработка робастной системы управления колонной атмосферной перегонки нефти *Восточно-европейский журнал передовых технологий*, 2015. Т. 5., №2 (77). С. 49-57.

References

- [1] R. Puerto, L.M. Jimenez and O. Reinoso, "Remote control laboratory via Internet using Matlab and Simulink," *Computer Applications in Engineering Education*, vol. 18, iss. 4, pp. 694-702, 2010.
- [2] Z. Janík and K. Zhakova, "Online design of Matlab/Simulink and SciLab/Xcos block schemes," *14th International Conference on Interactive Collaborative Learning, Piastany*, pp. 241-247, 2011.
- [3] L. Cirka and M. A. Kaluz, "Web-based Tool for Design of Simulink Models", *21st International Conference on Process Control*, pp. 92-97, 2017.
- [4] M. Sysel and M. Vaclavsky, "Remote Control and Monitoring in the Simulink," *International Journal of Circuits, Systems and Signal Processing*, vol. 7. iss. 2, pp. 83-90, 2013.
- [5] B. Bailey, *Connecting Simulink models to REST APIs*. URL: <https://billbailey.io/2020/07/10/connecting-simulink-models-to-rest-apis>.
- [6] L. Cirka, M. A. Kaluz and M. Fikar, "On-line remote control of MATLAB simulations based on asynchronous communication model," *Matlab conference in Institute of Chemical Technology, 2013, Prague*. URL: http://dsp.vscht.cz/konference_matlab/MATLAB13/prispevky/014_cirka.pdf
- [7] *Matsim: a sleek, intuitive interface for building Simulink models from a Matlab script*. URL: <https://github.com/gave92/Matsim>.
- [8] Stephan Brennan, *Write a Shell in C*. URL: <https://brennan.io/2015/01/16/write-a-shell-in-c/>
- [9] A. A. Stopakevich, "Mnogomernaja model' bragorektifikacionnoj ustanovki kosvennogo dejstvija," *Naukovi pratsi ONAKHT*, vol. 1, iss. 38, pp. 360-365, 2010.
- [10] A. A. Stopakevich, "Robust control system design of crude oil atmospheric distillation column," *Eastern-European Journal of Enterprise Technologies*, vol. 5., no. 2(77), pp. 49-57, 2015, DOI: <https://doi.org/10.15587/1729-4061.2015.50964>

УДК 004.42

ДОСЛІДЖЕННЯ МЕТОДІВ COMPUTER VISION ДЛЯ ВИКОРИСТАННЯ В АВТОМАТИЗОВАНИХ СИСТЕМАХ

Голубєв Л. П.¹, Ківа І. Л.², Матяш О. В.³

¹ Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, Україна

^{2,3} Київський національний університет технологій та дизайну, м. Київ, Україна

ORCID: ¹ <http://orcid.org/0000-0002-2980-8017>, ² <http://orcid.org/0000-0002-2920-5312>

E-mail: ¹golubev1@ukr.net, ²leonidovich1259@gmail.com, ³matasaleksandr97@gmail.com

Copyright © 2020 by author and the journal "Automation of technological and business – processes".

This work is licensed under the Creative Commons Attribution International License (CC BY).

<http://creativecommons.org/licenses/by/4.0>

DOI: <https://doi.org/10.15673/atbp.v12i4.1932>

Анотація. У статті виконано дослідження методів комп'ютерного стерео-зору за допомогою засобів бібліотеки OpenCV. Розглянуто також методи калібрування камер стерео-зору і побудови карти глибин дальності. На виробництві в технологічних процесах, в процесах контролю якості виробів, а також в роботизованих системах часто виникають завдання визначення відстані до об'єкта. Традиційні методи визначення відстані побудовані на базі використання ультразвукових і інфрачервоних датчиків. Однак, застосування цих пристроїв не забезпечує можливості розпізнавання образів. Тому в статті використано новий підхід до вирішення завдання визначення відстані до об'єкта на основі застосування стерео-зору. В якості візуальних елементів стерео-зору були використані WEB-камери Logitech C-170. Для визначення відстані до об'єкта за допомогою стерео-зору необхідно з початку виконати побудову карти глибини об'єктів. В якості програмних засобів, що забезпечують побудову карти глибини була обрана бібліотека з відкритим програмним кодом OpenCV. Використання цієї бібліотеки в некомерційних цілях є безкоштовною, проте при її використанні у розробників часто виникають труднощі у зв'язку неповним описом використовуваних функцій. Алгоритм побудови карти глибини складається з наступних кроків: калібрування камер відео-пари, усунення дисторсії зображень, ректифікація зображень і побудова карти глибини, на основі зображення з лівої і правої камер стерео-пари. Розроблена на базі міні-комп'ютера Orange-PI з використанням методів Computer Vision система побудови карти глибини сцени показала високу ефективність і дозволяє визначити відстань до об'єктів сцени.

Abstract. The article examines the methods of computer stereo vision using the OpenCV library. Methods of stereo cameras calibration and construction of a map of depths of range are also considered. In production in technological processes, in the processes of product quality control, as well as in automated systems, the task of determining the distance to the object often arises. Traditional methods of distance determination are based on the use of ultrasonic and infrared sensors. However, the use of these devices does not provide the ability to recognize images. Therefore, the article uses a new approach to solving the problem of determining the distance to the object based on the use of stereo vision. Logitech C-170 WEB cameras were used as visual elements of stereo vision. To determine the distance to the object using stereo vision, it is necessary to build a map of the depth of objects from the beginning. The OpenCV open source library was chosen as the software to build the depth map. Use of this library for non-commercial purposes is free, but when using it, developers often have difficulties due to the incomplete description of the functions used. The algorithm for constructing a depth map consists of the following steps: calibration of video pair cameras, image distortion elimination, image rectification and construction of a depth map, based on images from the left and right stereo pair cameras. Developed on the basis of the Orange-PI mini-computer using Computer Vision methods, the system of plotting the depth of the scene showed high efficiency and allows you to determine the distance to the objects of the scene.

Ключові слова: стерео-зір, Computer Vision, веб-камера, карта глибини

Keywords: stereo vision, Computer Vision, web-camera, range map

Вступ

Стерео-зір – створення зорового образу об'єкта, що спостерігається за рахунок обробки зображення об'єкта з різних сторін в предметному просторі. В реальний час воно широко використовується в технологічних процесах промислового виробництва, при контролі якості, а також в ігровій індустрії [3-5]. Однак особлива необхідність у використанні стерео-зору виникає в роботі мобільних роботизованих систем. Тому, що мобільний робот постійно повинен вирішувати задачу визначення свого положення в просторі, розпізнавання об'єктів, що знаходяться в його оточенні і визначення відстані до цих об'єктів.

Тому виникає завдання дослідження використання методів стерео-зору для вирішення перерахованих вище завдань.

Аналіз літературних даних і постановка проблеми

У 2001 р дослідники корпорації Intel оголосили про створення бібліотеки Open Source Computer Vision (OpenCV) для реалізації функцій стереоскопічного комп'ютерного зору [6, 7]. Вона являє собою інструментарій, який налічує понад 500 функцій обробки і аналізу зображень для створення додатків комп'ютерного зору, в тому числі засоби взаємодії людини з комп'ютером. Однак процедури стерео-зору, представлені в цій бібліотеці описані в літературі, на наш погляд, недостатньо повно. У роботах [1, 2] присутній короткий опис основних команд системи стерео-зору, однак, не описані докладно параметри використовуваних функцій, а вони є визначальними в якості роботи системи відео-зору мобільного робота.

Мета і завдання дослідження

Раніше традиційно для визначення дистанції мобільного пристрою до перешкоди в мобільних роботизованих пристроях використовувалися ультразвукові та інфрачервоні датчики. Вони досить добре справляються із завданням визначення відстані до об'єкта в діапазоні до 5м. Однак мобільні роботи оснащені цими датчиками зовсім позбавлені можливості розпізнавати об'єкти і, отже, адекватно оцінювати виникає ситуацію або слідувати за заздалегідь за програмованим маршрутом. Тому виникає завдання дослідження можливостей computer vision для використання в мобільних роботів. Метою дослідження є побудова карти глибин, використовуючи засоби бібліотеки OpenCV.

Результати досліджень

Карта глибин (depth map) - це зображення, на якому для кожного пікселя, замість кольору зберігається його відстань до камери.

В якості стереосистеми для мобільного робота використовувалися дві Web камери Logitech C-170, закріплені на твердій рівній поверхні і на фіксованій відстані одна від одної. Система стерео-зору була підключена до міні комп'ютера OrangePi, що працює під управлінням ОС Debian 10.

Принцип роботи системи заснований на побудові карти глибин представлений на рисунку 1.

Рис. 1 – Система стерео-зору на базі двох відео камер

Fig. 1 – Stereo vision system based on two video cameras

Для кожного пікселю картинки з лівої камери з координатами (x_0, y_0) здійснюється пошук відповідного пікселя на зображенні з правої камери. При цьому координати картинки з правої камери будуть відрізнятися від координат з лівої камери на величину зміщення d тобто $(x_0 - d, y_0)$. Пошук відповідного пікселя виконується шляхом обчислення максимуму функції відгуку, в якості якої зазвичай використовують кореляцію околиць пікселів.

Тоді залежність між зміщенням і глибиною пікселів можна виразити наступною формулою:

$$Z = \frac{f \cdot T}{d}, \quad (1)$$

де Z – відстань до пікселя; f – фокусна відстань камери; T – відстань між камерами; d м зміщення пікселя.

Для отримання даних про глибину об'єкта на сцені за допомогою зображень, одержуваних при обробці відео-потoku від двох камер необхідно виконати калібрування камер і далі побудувати карту глибин на основі формули (1).

У OpenCV для калібрування використовується метод шахівниці (ChessBoards) (рис.2). Як об'єкт калібрування в цьому методі використовується лист з чорними і білими квадратами відомого розміру. Перевагою методу є: відсутність пріоритету до горизонтальної або вертикальної осі [6].

Рис. 2 – Приклад шаблону шахівниці

Fig. 2 – Example of a chessboard template

Цей метод реалізований в OpenCV функцією:

```
int cvFindChessboardCorners (const void* image, CvSize pattern_size, CvPoint2D32f* corners),
```

де *image* – зображення, що містить шахову дошку; *pattern_size* – показує, скільки кутів у кожному рядку і стовпці дошки; *corners* – покажчик на масив, в якому можуть бути записані розташування кутів.

У разі успішного виконання (функція успішно знайшла всі кути), повертається значення що буде кількістю кутів шахівниці, в іншому випадку - як результат видається 0. Далі два зображення зберігаються в окремих змінних структури `IplImage`. Результатом виконання даного кроку є усунення будь-яких дефектів у зображеннях, які виходять з камер.

Далі виконується калібрування камер стереопари за допомогою бібліотечної функції OpenCV `calibrate Camera ()`. Після цього проводиться калібрування всієї стереопари. Дані, отримані на попередньому кроці подаються на вхід бібліотечної функції OpenCV `stereo Calibrate ()`, яка розраховує внутрішні параметри стереопари. Після процедури стерео калібрування ми отримуємо фокусну відстань камер в пікселях і координати принципів точок камер.

Розглянемо більш докладно побудову карти диспарантностей.

Для кожного пікселя лівої картини з координатами (x_0, y_0) виконується пошук пікселя на правій картинці. При цьому передбачається, що піксель на правій зображенні повинен мати координати $(x_0 - d, y_0)$, де d - невідповідність / зміщення (disparity) (рис. 3).

Disparity задає кількість точок, сегментованих по площинах. Площині розташовані на різній відстані від камер (Cameras). Пошук відповідного пікселя виконується шляхом обчислення максимуму функції відгуку, в якості якої може виступати, наприклад, кореляція околиць пікселів. В результаті виходить карта зсувів (disparity map).

Рис. 3 – Побудова карти зсувів
Fig. 3 – Construction of a map of landlides

Далі по парі відповідних точок, знайдених раніше, можна визначити координати образу в тривимірному просторі. Для цього використовуємо функцію:

`cvReprojectImageTo3D (const CvArr* disparity, CvArr* _3dImage, const, CvMat* Q),`

де `disparity` – карта відмінностей; `_3dImage` – масив, що містить тривимірне зображення; `Q` – матриця перебудовувань, яка виходить після калібрування матриці і містить параметри:

$$Q = \begin{bmatrix} 1 & 0 & 0 & -cx \\ 0 & 1 & 0 & -cy \\ 0 & 0 & 0 & f \\ 0 & 0 & -\frac{1}{T} & (cx - cx')/T \end{bmatrix} \quad (2)$$

де T – відстань між камерами по осі X (передбачається, що координати Y однакові); f – фокусна відстань; cx , cy – координати точки на головному (центральному) промені лівої камери; cx' – x координата на правому зображенні лівої точки (це робиться для того, що уточнити, як пов'язана відстань між камерами і кількість пікселів)

Для ілюстрації програмної реалізації побудови матриці диспарантності далі наводиться фрагмент програми мовою C++:

```
// створення об'єкта для лівої камери
VideoCapture leftCam(0); //lets say 0 is left, 1 is right
if (leftCam.isOpened() == false){cout << "error: Webcam connect unsuccessful\n";
return(0); }
// створення об'єкта для правої камери
VideoCapture rightCam(1); //lets say 0 is left, 1 is right
if (rightCam.isOpened() == false){cout << "error: Webcam connect unsuccessful\n";
return(0); }
```

```
// створення матриці зображень лівої і правої камери
Mat left, right;
Mat leftClone, rightClone;
char charCheckForEscKey = 0;
while ( charCheckForEscKey != 27 && leftCam.isOpened() )
{
// читання зображення з лівої камери
leftCam.read(left);
if (left.empty()){cout << "No frame to read" << endl; break;}
leftClone = left.clone(); //copy from the left camera
// запис зображення з лівої камери в файл
imwrite("left.png", leftClone); // write it to screenshot.png in this
directory
// читання зображення з правої камери
rightCam.read(right);
if (right.empty()){cout << "No frame to read" << endl; break;}
rightClone = right.clone(); //copy from the left camera
// запис зображення з правої камери в файл
imwrite("right.png", rightClone); // write it to screenshot.png in this
directory
// формування матриці зображення лівої камери
Mat im_left = imread("left.png"); //left cam picture
// формування матриці зображення правої камери
Mat im_right = imread("right.png"); // right cam picture
Size imagesize = im_left.size();
// формування матриці диспарантності лівої камери
Mat disparity_left= Mat(imagesize.height,imagesize.width,CV_16S);
// формування матриці диспарантності правої камери
Mat disparity_right=Mat(imagesize.height,imagesize.width,CV_16S);
Mat g1,g2,disp,disp8;
// перетворення зображення лівої камери в градації сірого
cvtColor(im_left,g1, COLOR_BGR2GRAY);
// перетворення зображення правої камери в градації сірого
cvtColor(im_right,g2, COLOR_BGR2GRAY);
// створення об'єкта стерео-зору
Ptr<cv::StereoBM> sbm = StereoBM::create(0,21);
// установка параметрів об'єкта StereoBM
sbm->setDispl2MaxDiff(1);
sbm->setSpeckleRange(8);
sbm->setSpeckleWindowSize(9);
sbm->setUniquenessRatio(0);
sbm->setTextureThreshold(507);
sbm->setMinDisparity(-39);
sbm->setPreFilterCap(61);
sbm->setPreFilterSize(5);
sbm->compute(g1,g2,disparity_left);
normalize(disparity_left, disp8, 0, 255, NORM_MINMAX, CV_8U);
// виведення зображення лівої камери у вікні «Left»
namedWindow("Left", WINDOW_AUTOSIZE);
imshow("Left", im_left);
// виведення зображення правої камери у вікні «Right»
namedWindow("Right", WINDOW_AUTOSIZE);
imshow("Right", im_right);
// виведення зображення карти глибини в вікні «Depth map»
namedWindow("Depth map", WINDOW_AUTOSIZE);
imshow("Depth map", disp8);
// виведення клонованого зображення лівої камери у вікні «Left Cloned»
namedWindow("Left Cloned", WINDOW_FREERATIO);
imshow("Left Cloned", leftClone); // left is the left pic taken from camera 0
charCheckForEscKey = waitKey(1);
}
return(0);
}
```

Процес побудови карти глибин з використанням стерео-зору наведено на рисунках 4-6.

Рис. 4 – Вихідні зображення з лівої і правої камер відповідно
Fig. 4 – Input images from the left and right cameras, respectively

Рис. 5 – Слайдери для управління параметрами
об'єкту StereoBM

Fig. 5 – Sliders to control the parameters
of the StereoBM object

Рис. 6 – Побудована карта глибини
Fig. 6 – Constructed depth map

Основні параметри, об'єкта StereoBM, що налаштовуються:

minDisparity – мінімально можливе значення диспаратності. Зазвичай цей коефіцієнт дорівнює нулю, але іноді алгоритми випрямлення можуть зрушувати зображення, тому цей параметр необхідно відповідним чином налаштувати;

numDisparities – різниця між максимальною і мінімальною невідповідністю. Значення цього параметра завжди більше нуля. У поточній реалізації цей параметр повинен бути кратний 16;

blockSize – розмір блоку. Це повинно бути непарне число ≥ 1 . Зазвичай цей параметр повинен бути в діапазоні від 3 до 11;

P1 – перший параметр, керуючий гладкістю диспаратності;

P2 – другий параметр, керуючий гладкістю диспаратності. Чим більше ці значення, тим більше гладким є невідповідність;

disp12MaxDiff – максимально допустима різниця (в цілочисельних піксельних одиницях) при перевірці невідповідності зліва направо. Для відключення перевірки треба встановити негативне значення;

preFilterCap – значення усічення для попередньо відфільтрованих пікселів зображення. Алгоритм спочатку обчислює x-похідну в кожному пікселі і обрізає її значення з інтервалом $[-preFilterCap, preFilterCap]$. Отримані значення передаються в функцію вартості пікселів Берчфілд-Томасі.

З огляду на величину освітленості об'єктів в кадрі, завантаженості сцени і її зашумленості для отримання найкращого результату необхідно виконати настройку основних параметрів об'єкта StereoBM, змінюючи положення відповідних слайдерів.

Висновки

Розроблена стереосистема здатна визначати відстань до об'єктів, використовуючи стереоскопічний ефект в режимі реального часу. Застосування такої системи на мобільному роботизованому пристрої дозволяє підвищити безпеку його пересування. Стереосистема складається з блоку управління і двох ширококутних Web- відеокамер. Для визначення відстані в більш широкому куті огляду можна використовувати кілька стереопар, розташовуючи їх, наприклад, таким чином, щоб отримати повний тілесний кут.

Список використаних джерел

- [1] Bradski Gary, Learning OpenCV 3: Computer vision in C++ with the OpenCV library. Adrian Kaehler, Gary Bradski. O'Reilly Media -2017, 1018 p.
- [2] Глория Буэно Гарсия Обработка изображений с помощью OpenCV//Глория Буэно Гарсия, Оскар Дениз Суарес, Хосе Луис Эспиноса Аранда, Хесус Салидо Терсеро, Исмаэль Серрано Грасиа, Ноэлия Валлез Энано, пер. с англ. Слинкин А. А. – М.: ДМК Пресс, 2016. – 210 с.: ил.
- [3] OpenCV Open Source Computer Vision URL: <http://opencv.org> (дата обращения: 28 мая 2016)
- [4] Gary Bradski, Adrian Kaehler Learning OpenCV // O'Reilly Media, 2008, P. 580
- [5] Batavia P., Singh S. Obstacle detection using adaptive color segmentation and color stereo homography // IEEE International Conference on Robotics and Automation. - 2001. - Vol. 1. - P. 705-710.
- [6] Tsai L., Hsieh J., Fan K. Vehicle detection using normalized color and edge map // 18th IPPR Conference on Computer Vision, Graphics and Image Processing. - Taipei, 2005. - P. 849-856.
- [7] Fleyeh H. Road and traffic sign color detection and segmentation - A Fuzzy Approach // IAPR Conference on Machine Vision Applications. - Tsukuba Science City, 2005. - P. 124-127.
- [8] Fuzzy-based algorithm for color recognition of license plates / F. Wang, L. Man, B. Wang, Y. Xiao, W. Pan, X. Lu // Pattern Recognition Letters. - 2008. - Vol. 29. - P. 1007-1020.

References

- [1] A. Kaehler and G. Bradski, *Learning OpenCV 3: Computer vision in C++ with the OpenCV library*. O'Reilly Media, 2017, 1018 p.
- [2] G. Bueno Garsiya et al, *Obrabotka izobrazheniy s pomoshchyu OpenCV*, per. s angl. Slinkin A., Moscow, DMK Press, 2016, 210 p.
- [3] *OpenCV Open Source Computer Vision* (2016, May 28) [Online]. Available: <http://opencv.org>
- [4] G. Bradski and A. Kaehler, *Learning OpenCV*, O'Reilly Media, 2008, 580 p.
- [5] P. Batavia and S. Singh, "Obstacle detection using adaptive color segmentation and color stereo homography," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2001, pp. 705-710.
- [6] L. Tsai, J.Hsieh and K. Fan, "Vehicle detection using normalized color and edge map," in *18th IPPR Conference on Computer Vision, Graphics and Image Processing*, Taipei, 2005, pp. 849-856.
- [7] H. Fleyeh, "Road and traffic sign color detection and segmentation - A Fuzzy Approach," in *IAPR Conference on Machine Vision Applications*, Tsukuba Science City, 2005, pp. 124-127.
- [8] F. Wang et al, "Fuzzy-based algorithm for color recognition of license plates," *Pattern Recognition Letters*, vol. 29, pp. 1007-1020, 2008.

УДК 004.41

ІНФОРМАЦІЙНО-АНАЛІТИЧНА СИСТЕМА ПРИЙМАЛЬНОЇ КОМІСІЇ ОНАХТ ЯК ОСНОВА АВТОМАТИЗОВАНОГО УПРАВЛІННЯ ФОРМУВАННЯ КОНТИНГЕНТУ СТУДЕНТІВ

Мороз А. М.¹, Похлебіна Н. О.², Хобін В. А.³

Одеська національна академія харчових технологій, м. Одеса, Україна

ORCID: ¹ <http://orcid.org/0000-0002-6655-3578>, ² <http://orcid.org/0000-0002-9789-608X>,

³ <http://orcid.org/0000-0003-0238-8371>

E-mail: ¹andrey.work2001@gmail.com, ²pohlebina.odessa@gmail.com, ³khobin@onaft.edu.ua

Copyright © 2020 by author and the journal "Automation of technological and business-processes".

This work is licensed under the Creative Commons Attribution International License (CC BY).

<http://creativecommons.org/licenses/by/4.0>

DOI: <https://doi.org/10.15673/atbp.v12i4.1933>