PERFORMANCE ANALYSIS OF HECTOR SLAM AND GMAPPING FOR NAVIGATION FOR MOBILE ROBOT NAVIGATION

[1]Paulus Sakti Laksono, [2]Tubagus Maulana Kusuma
[1,2]Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Gunadarma
[1,2]Jl. Margonda Raya No. 100, Depok 16424, Jawa Barat
[1]paulus.sakti@gmail.com, [2]mkusuma@staff.gunadarma.ac.id

Abstrak

Pemetaan merupakan salah satu fitur penting yang harus dimiliki oleh sebuah robot yang bergerak. Kemampuan robot mengenali keberadaan sekitarnya dan menerjemahkannya ke dalam bentuk peta memungkinkan robot untuk bernavigasi dari suatu titik ke titik lain dengan efektif dan mampu menghindari hambatan yang terjadi saat proses navigasi. Algoritma SLAM sudah menyediakan fitur pemetaan dan kemudian mampu melokalisasi posisi terhadap peta terus-menerus. Pada eksperimen ini data pemindaian laser 2D diperoleh menggunakan RPLidar-A1 kemudian diproses oleh algoritma slam yakni gmapping dan hector mapping sehingga menghasilkan peta grid okupansi. Peta tersebut ditampilkan oleh widget visualisasi RViz, dan diukur besaran panjangnya menggunakan measurement tools RViz. Hasil peta grid okupansi yang dihasilkan oleh algoritma slam dengan laser scan matcher memiliki banyak noise, dan kehilangan orientasi. Pada besaran panjang lokasi yang diukur, algoritma gmapping memiliki error sedikit lebih banyak. Algoritma hector mapping memiliki kinerja yang lebih baik dibandingkan gmapping dengan laser scan matcher pada perangkat RPLidar-A1.

Kata Kunci*: Arduino, Encoder, LIDAR, Robot Operating System, Raspberry Pi, SLAM.*

Abstract

One of the most significant elements of a moving robot is mapping. The robot's capacity to identify its surroundings and translate them into a map allows it to navigate effectively from one spot to another while avoiding impediments that may arise during the navigation process. The SLAM method already has a mapping capability, so it can continuously localize the position against the map. In this experiment, 2D laser scanning data was obtained using RPlidar-A1 and then processed by the slam algorithm, namely gmapping and hector mapping to produce an occupancy grid map. The map is displayed by the RViz visualization widget, and its length is measured using the RViz measurement tools. The results of the occupancy grid map generated by the gmapping algorithm with a laser scan matcher have a lot of noise, and lose orientation. At the measured point length, the gmapping algorithm has slightly more error. The hector mapping algorithm has better performance than gmapping with a laser scan matcher on the RPLidar-A1 device.

Keywords*: Arduino, Encoder, LIDAR, Robot Operating System, Raspberry Pi, SLAM.*

INTRODUCTION

The fourth industrial revolution builds on the digital revolution, representing new ways in which technology becomes embedded in society and even the human body. The fourth industrial revolution was marked by the emergence of technological breakthroughs in a number of fields, including robotics, artificial intelligence, nanotechnology, quantum computing, biotechnology, internet of things, 3D printing, and autonomous vehicles [1]. The application of robots by industry in the world continues to increase from the previous year and is predicted to continue to grow. According to data provided by Deloitte Touche Tohmatsu in 2019, there was a significant growth in numbers from 2016 to 2019 with an average increase of 300 thousand units of robot use for industry. It has also been predicted that there will be a worldwide demand for 3.5 million robot units for industry by 2021[2].

Robots that are commonly used nowadays can be operated manually by humans using a remote control or they can move autonomously using the line following approach (following the lines that have been made previously). Manually operated robots are inefficient at their jobs, while robots that use the line following approach can only move along the lines that have been drawn. As a result, the mobile robot will require another autonomous navigation system in order to travel freely in the inside environment [3].

There will be a lot more interaction between humans and robots, as well as robots with other robots, as the industry adopts robotics. This will present issues if the robot is placed in a dynamic environment with constantly moving items and humans. The robot must recognize the surrounding environment, which may include things or obstacles that have never been seen or identified before. There are a variety of methods and sensors that can be employed, including LIDAR, which has been used in the automotive industry. With the help of the SLAM algorithm, LIDAR technology can map and locate positions. The robot can use this approach to map unknown surroundings. The robot's capacity to map and localize itself enables it to navigate from one spot to another effectively and efficiently, reducing the hazards associated with its operation. The SLAM system also enables the robot to select the shortest path and track it..

The more accurate the map generated by the SLAM algorithm, the better the robot will navigate. This research will look at how to use the SLAM algorithm to help the robot map and localize its position so that it can move in accordance with the current environment. Our aim was to build an autonomous navigation platform for indoor application. This robot supports mapping and localization functions via RPLidar-A1 applications as inputs. In this paper, we compare the map generated by the robot model to the destination to determine the accuracy of a SLAM (Simultaneous Localization and Mapping) based robot model implemented in ROS (Robot Operating System). The map results are then compared, and the length is measured using the RViz measurement tool to see the value closest to the original.

The ROS operating system has been used in a number of SLAM studies, one of which is the development of soccer robots. This robot consists of various systems, namely, servo controller, vision system, transmitter, and receiver for strategy instruction. For the robot to accomplish the task of playing football, all these sophisticated systems must be integrated. The basic functions that a robot requires to be able to play talk ball include moving the robot, sensing the presence of the ball, and localizing the robot's position in relation to the environment. This is where ROS comes in, assisting in the integration of the entire system so that the robot can perform complex tasks like moving closer to the ball identified by the camera [4].

Rajesh Kannan Megalingam, Chinta Ravi Teja, Sarath Sreekanth, and Akhil Raj conducted additional study on the application of SLAM on the Gazebo simulation tool. They utilise the SLAM method, which is used to map and navigate an indoor environment by a moving robot. The efficacy of the robot utilizing the SLAM algorithm is calculated in this study by measuring the robot's travel time when navigating to specific points / locations. RViz, which had been given an impediment object, was used to conduct the experiment virtually. The gazebo robot simulator enables us to generate complicated environments so that people may get an idea of how the robot will perform in real-world scenarios [5].

Another study on mapping and localization using SLAM was conducted by SangYoung Park and GuiHyung Lee. They developed a control system device built on the ROS framework. This control system manages the mapping and localization of 2 robots working together in an unknown area. SLAM technology is used to determine the position of the robot and its surroundings. Windows programs written in C# programming language are used as interfaces for the communication of the two robots and are linked to the ROS framework. This study does not use sensors other than for self-driving and controlling multi-robots. This system uses 1 PC which acts as a server and Odroid-U3 on a robot that handles sensors and communication to the master [6].

This paper is organized as follows. In Section 2, we discuss the software, hardware, and prototype design of a mobile picking robot prototype. In Section 3, we generate experimental data for compared mapping algorithms. Section 4 brings this paper to a close.

METHODOLOGY

This chapter will discuss the design of a mobile picking robot prototype, starting with the software, hardware, and prototype design. The software will explain how to read the encoder, from how to read the encoder for each wheel to how the wheel is run using the velocity topic command published by the Teleop Keyboard application through the Robot Operating System. Then proceed with the discussion of the map simulation process generated by the robot prototype on the RViz application. Hardware design is used to determine the needs of what devices are included in the prototype of the mobile picking robot. Hardware diagrams and wiring will also be discussed and divided based on each input / output used. An overview of the system in general can be seen below.

We can see in the Figure 1 that the hardware that composes the robot prototype is divided into input, process, and output blocks. LIDAR and Rotary Encoder function as inputs to this system, both of which have the same goal, which is to enable robots to perform simultaneous localization and mapping. Even so, these two devices have different functions. The rotary encoder will oversee determining how far the robot is, which then provides the information to the Arduino device, while the LIDAR functions to map the surrounding conditions of the robot, and the information will be handled by the Raspberry Pi.

In the process block, there is communication that occurs between the Arduino and the Raspberry Pi. This communication aims to send information about the speed generated by the velocity topic command on the ROS network to Arduino, while Arduino also provides speed information generated by the encoder to the ROS network. Connecting Arduino and Raspberry Pi directly via serial communication allows Arduino to publish and subscribe to the ROS network.

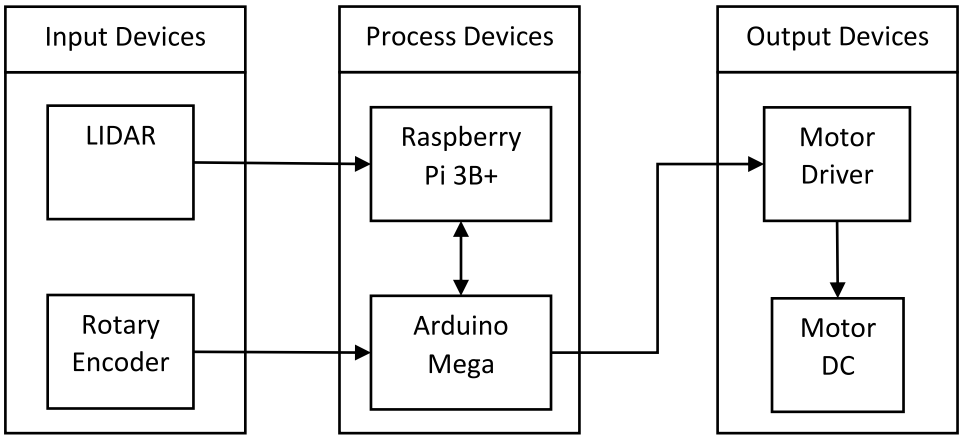


Figure 1. General Research Overview

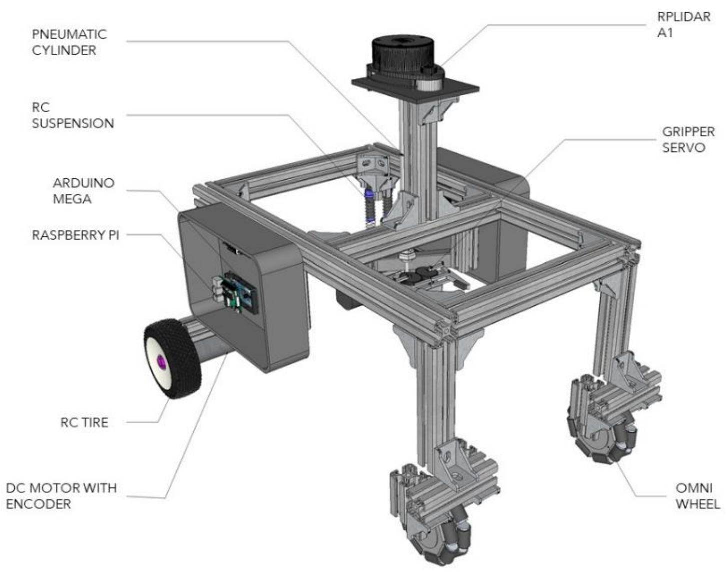


Figure 2. Component design and placement

In this prototype, the only hardware that is output is a DC motor. The DC motor, of course, serves to move the prototype robot from one position to another. This motor is controlled by the BTS7960 motor driver, which is capable of flowing current up to 43 amperes. A simple h-bridge circuit controlled via Arduino's digital pins.

Prototyping and component placement of the mobile picking robot can be seen in the image above. There are 2 omni wheels on the front side of the robot. This wheel selection is due to the characteristics of the wheels that can move in all directions without requiring complex control. Then on the back side, there is an RC tire mounted on a DC motor. Meanwhile, components such as Arduino and Raspberry Pi, along with a buck converter, are attached to the sides of the robot using a box. Lidar itself is installed above the pneumatic so as not to interfere with the scanning process reflected from the pneumatic valve.

Simultaneous Localization and Mapping

An autonomous robot should be able to explore its surroundings safely without bumping into people or objects. Simultaneous localization and mapping (SLAM) allow the robot to do this by first knowing how the conditions of the surrounding environment are using mapping and then knowing where the robot is located in the surrounding environment or on a map with localization [7]. The SLAM algorithm can be applied to 1D, 2D, and 3D sensors. Various examples include acoustic sensors, laser distance sensors, stereo image sensors, and so on. Different types of sensors used in the implementation of SLAM will result in various errors when the robot tries to estimate the movement or location [8]. The biggest challenge of the SLAM algorithm is determining whether the measurement sensor picks up data at the same point at the same time compared to the same object in the real environment.

To understand the slam algorithm more simply, we can look at the Figure 3. At the initial stage, the algorithm will start the scanning process and estimate the initial position of the robot. The robot will move a little in a short time, and then the laser will perform the scan. At the second scan, we get a new scan result and estimate. This scan is then compared with the map created to estimate the change in position. This takes several times to update the map and approximate position of the robot.

There are several 2D SLAM algorithms that have been implemented on ROS, namely Gmapping, Hector SLAM, Karto SLAM, coreslam, and Lago SLAM. Gmapping, Hector SLAM, and Karto SLAM have the best performance among other SLAM algorithms, and in this study, Hector SLAM is the choice without using odometry [9]. This algorithm also does not require roll or pitch movement from the sensor. This makes Hector SLAM the right choice and makes it easy to apply to this robot prototype.

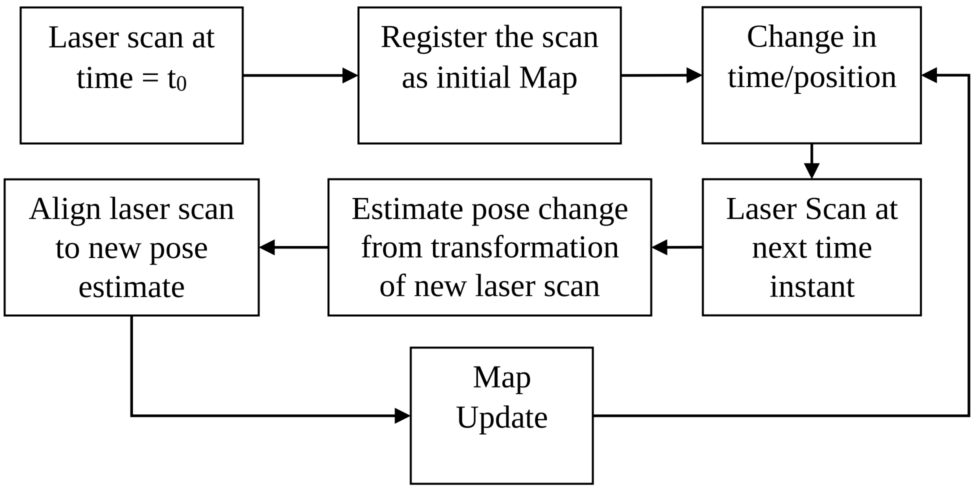
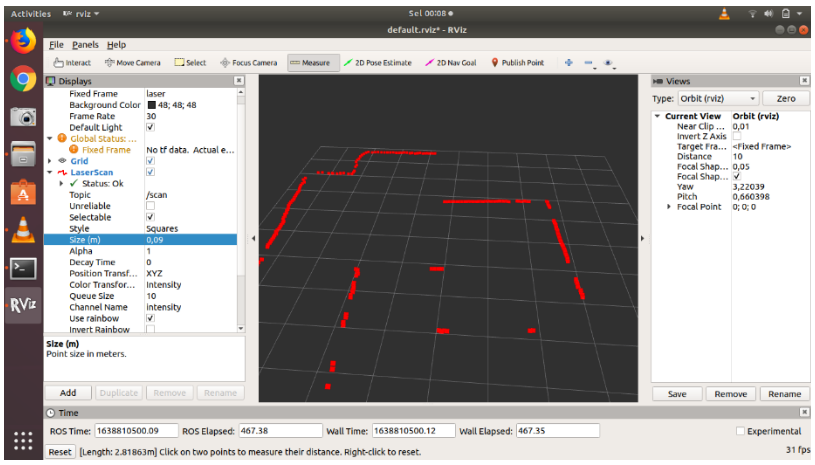

Laser scan at time = t_0
Register the scan as initial Map
Change in time/position
Laser Scan at next time instant
Estimate pose change from transformation of new laser scan
Align laser scan to new pose estimate
Map Update

Figure 3. Overview of SLAM



Fixed Frame	laser
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
Global Status: ...	
Fixed Frame	No tf data. Actual e...
Grid	✓
LaserScan	✓
Status: Ok	
Topic	/scan
Unreliable	☐
Selectable	✓
Style	Squares
Size (m)	0,09
Alpha	1
Decay Time	0
Position Transf...	XYZ
Color Transfor...	Intensity
Queue Size	10
Channel Name	intensity
Use rainbow	✓
Invert Rainbow	☐

Size (m)
Point size in meters.

Type: Orbit (rviz)

Current View	**Orbit (rviz)**
Near Clip ...	0,01
Invert Z Axis	☐
Target Fra...	<Fixed Frame>
Distance	10
Focal Shap...	0,05
Focal Shap...	✓
Yaw	3,22039
Pitch	0,660398
Focal Point	0; 0; 0

ROS Time: 1638810500.09 ROS Elapsed: 467.38 Wall Time: 1638810500.12 Wall Elapsed: 467.35

[Length: 2.81863m] Click on two points to measure their distance. Right-click to reset.

Figure 4. Preview scan topic on RViz

The image above shows the raw scan results produced by the rplidarNode program. The program generates a "scan" topic and uses LaserScan messages. This data will then be continued as input to the hector_mapping program and generate several topics such as map_metadata, map, slam_out_pose, and poseupdate. The hector_mapping program uses the tf or transform application for scanning data transformation, thus allowing the LIDAR device to change places, not according to what has been defined in the base frame. The RViz application can immediately display the mapping results and display them in real time.

EXPERIMENTAL RESULTS

Simultaneous localization and mapping experiment using the Gmapping and Hector SLAM algorithm and carried out in an indoor room with a flat floor surface. This indoor environment has several rooms such as bedrooms, kitchens, living rooms, family rooms and bathrooms. Details of the size of each room are described in the room plan in Figure. This experiment will see how the map results generated by the gmapping and hector_mapping algorithm correspond to the original environment.

The result of the map formed by the hector_mapping program were drawn on the RViz visualization application. Even though it does not use odometry, this algorithm is able to map the room quite well, it can be seen on the upper side, the bedroom to the room has an asymmetrical shape, according to the actual situation. In contrast to hector_mapping which does not require odometry, gmapping requires assistance from laser_scan_matcher to provide the odometry data because this algorithm requires a data transform to determine whether to enter a new scan and as an initial guess for matching internal 2D scans. The result of the map shown below, we can see the structure poles in the living room and stairs are well illustrated.

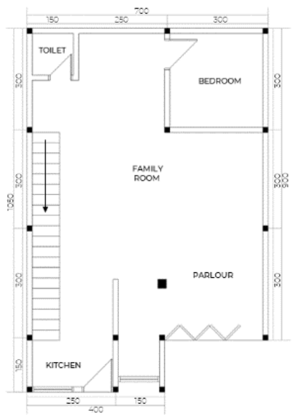


Figure 5a. Indoor Floor Plan

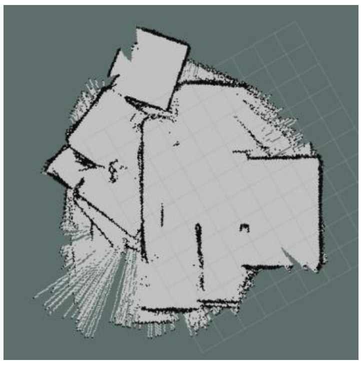


Figure 6b. Map generated by gmapping

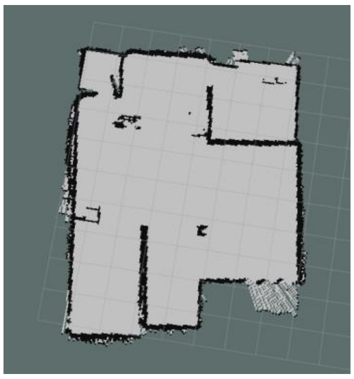


Figure 7c. Map generated by hector_slam

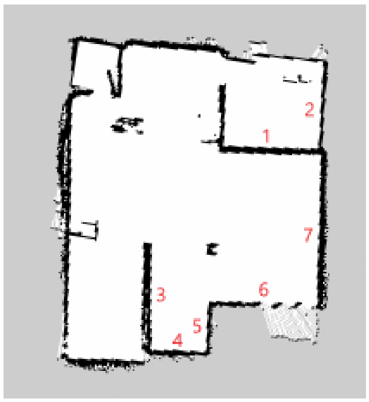


Figure 8. Map with predefined number

Table 1. Size Comparison with Rviz Measurement Tool

No	Measurement (cm)			Error	
	hector_mapping	gmapping	Tape	hector_mapping	gmapping
1	278	277	281	3	4
2	261	255	249	12	6
3	303	338	317	14	21
4	167	148	171	4	23
5	143	149	149	6	0
6	305	304	316	11	12
7	435	450	440	5	10
Mean Error				7,86	10,86

After the floor plan has been formed, we can use the tool in the RViz visualization application, namely the measurement tool. This feature can calculate the distance between 2 points. We can assign values to both points by pressing the coordinate points displayed in the /map frame. After getting the two desired values to be calculated, the program will immediately display the measurement results in meters. Figure 6. shows a map that has been numbered according to each location. This experiment will see how accurate the measurement results are with the RViz measurement tool and manual measurements using a measurement tape. In the first and second experiments carried out in the bedroom, the values were 278 and 261. Meanwhile, the readings with the measurement tape got the values 281 and 249, so the error or difference between the readings through the RViz measurement tool and manual measurements was 3 and 12. The same was done at different locations marked with experimental numbers 2 to 7, the average error was 7.86 cm. This shows that the hector_mapping algorithm without using odometry can map the room with an error of 7.86 cm.

Meanwhile, by using the map generated by the gmapping algorithm, we get readings of 277 and 255cm, in the bedroom. Likewise with the living room, at the locations shown in numbers 3 to 5, the results obtained are 338, 148, and 149cm. The results given by the gmapping algorithm in combination using laser_scan_matcher is not too different in distance measurements, although the hector algorithm has a smaller error of 3cm in Table 6. However, the results of the occupancy grid map formed by the gmapping algorithm, there is a lot of noise, and shift of the odometry data generated by the laser_scan_matcher. This is due to the use of laser_scan_matcher which replaces odometry data.

CONCLUSION AND SUGGESTION

After performing several assignments such as designing, constructing, and testing, the study results could be defined. On the basis of the outcomes of the previous studies, it can be concluded that the design and manufacture of the tools work correctly and in accordance with the requirements. The design has been successful based on the objectives made, using the slam algorithm to build maps that are suited to the surroundings. In this research, the hector slam algorithm outperformed the gmapping technique, which relies on laser scan matcher instead of odometry data. To improve map reading accuracy, odometry from devices such as encoders, inertia measuring units, and GPS is required. Because there is a mismatch between the encoder readings and the parameters given, manually checking the encoder readings is necessary.

BIBLIOGRAPHY

[1] M.A. Purba, and A.D. Yando, *Revolusi Industri 4.0*, Batam : CV BATAM PUBLISHER, 2020.

[2] Y. Pusparisa, Penggunaan Robot Industri Diprediksi Naik 96% dalam 5 Tahun, Databoks Katadata, 2019.

[3] T.A. Putra, M. Muliady, D. Setiadikarunia, "Navigasi Indoor Berbasis Peta pada Robot Beroda dengan Platform Robot Operating System", *Jetri J. Ilm. Tek. Elektro*, vol. 17, pp. 121–144, 2020.

[4] S. Susanto, J. Suroto, R. Analia, "The ROS: Kinetic Kame for Humanoid Robot BarelangFC", *J. Integr*, vol.13, pp. 68–77, 2021.

[5] R. Kannan Megalingam, C. Ravi Teja, S. Sreekanth, A. Raj, " ROS based Autonomous Indoor Navigation Simulation Using SLAM Algorithm", *Int. J. Pure Appl. Math*, vol.118, pp. 199–205, 2018.

[6] S. Park, and G. Lee, " Mapping and localization of cooperative robots by ROS and SLAM in unknown working area", *in: 2017 56th Annu. Conf. Soc. Instrum. Control Eng. Japan*, 2017: pp. 858–861.

[7] S. Thrun, and J.J. Leonard, *Simultaneous Localization and Mapping*, in: B. Siciliano, O. Khatib (Eds.), Springer Handb. Robot., Springer Berlin Heidelberg, Berlin, Heidelberg, 2008: pp. 871–889.

[8] D. Huang, Z. Cai, Y. Wang, X. He, "A real-time fast incremental SLAM method for indoor navigation", *in: 2013 Chinese Autom. Congr.*, 2013: pp. 171–176.

[9] P. Beinschob and C. Reinke, "Graph SLAM based mapping for AGV localization in large-scale warehouses", *in: 2015 IEEE Int. Conf. Intell. Comput. Commun. Process.*, 2015: pp. 245–248.