International Journal of Control, Automation and Systems 14(6) (2016) 1572-1581
http://dx.doi.org/10.1007/s12555-014-0553-y
ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Positioning and Obstacle Avoidance of Automatic Guided Vehicle in Partially Known Environment

Pandu Sandi Pratama, Trong Hai Nguyen, Hak Kyeong Kim, Dae Hwan Kim, and Sang Bong Kim*

Abstract: This paper presents positioning and obstacle avoidance of Automatic Guidance Vehicle (AGV) in partially known environment. To do this task, the followings are done. Firstly, the system configuration of AGV is described. Secondly, mathematical kinematic modeling of the AGV is presented to understand its characteristics and behavior. Thirdly, the Simultaneous Localization and Mapping (SLAM) algorithm based on the laser measurement system and encoders is proposed. The encoders are used for detecting the motion state of the AGV. In a slippery environment and a high speed AGV condition, encoder positioning method generates big error. Therefore, Extended Kalman Filter (EKF) is used to get the best position estimation of AGV by combining the encoder positioning result and landmark positions obtained from the laser scanner. Fourthly, to achieve the desired coordinate, D* Lite algorithm is used to generate a path from the start point to the goal point for AGV and to avoid unknown obstacles using information obtained from laser scanner. A backstepping controller based on Lyapunov stability is proposed for tracking the desired path generated by D* Lite algorithm. Finally, the effectiveness of the proposed algorithms and controller are verified by using experiment. The experimental results show that the AGV successfully reaches the goal point with an acceptable small error.

Keywords: Automatic guided vehicle, differential drive, localization, obstacle avoidance, tracking.

1. INTRODUCTION

The use of AGV is the one of the most preferred means to reduce the operation costs by helping the factories to automate a manufacturing facility or warehouse. The common challenging problems related with AGV operation are positioning, path planning, obstacle avoidance and trajectory tracking.

The commonly used positioning methods in AGV were visual line follower using camera sensor [1], inductive guidance using electrical wire buried under the floor [2], semi-guided navigation method by using magnetic tapes [3], wall following algorithm [4] and laser navigation system [5]. However, these algorithms depend on predetermined paths or landmark positions. Therefore, they can only work in known environments.

To generate the optimal path from the start to goal position using a given map, rapidly-exploring random trees algorithm [6], potential function [7] and gradient method (GM) [8] were proposed. Those algorithms only work for known and static environments.

In factory environments, a sudden arrival of obstacle could block the AGV path. To deal with this, several obstacle avoidance algorithms were proposed such as curvature velocity method [9], dynamic window approach [10], moving obstacle avoidance [11], and obstacle avoidance based on obstacle geometric [12]. However, as the calculated area are limited, the goal position reachability condition is not guaranteed.

After an optimal path is generated, a trajectory tracking control algorithm is need for the AGV to track the optimal path. Several control algorithms are proposed to accomplish this task such as Fuzzy-PID [1], sliding mode control theory [13], and time varying feedback control law [14]. Among these controllers, although the stability of the system is guaranteed, it might not be easy to find an appropriate control law.

To solve these problems, this paper proposes a new algorithm for positioning, path planning, obstacle avoidance and trajectory tracking in partially known environment. Positioning of AGV is obtained by using SLAM algorithm based on EKF. For path planning and obstacle avoidance, D* Lite algorithm based on a given map and laser scanner data is proposed. To guarantee the tracking errors to be-

Manuscript received December 15, 2014; revised October 8, 2015; accepted December 29, 2015. Recommended by Associate Editor Dong-Joong Kang under the direction of Editor Fuchun Sun. This research was supported by a grant (11 Transportation System- Logistics 02) from Transportation System Efficiency Program funded by Ministry of Land, Infrastructure and Transport (MOLIT) of Korean government.

Pandu Sandi Pratama, Trong Hai Nguyen, Hak Kyeong Kim, Dae Hwan Kim, and Sang Bong Kim are with Department of Mechanical Design Engineering, Pukyong National University, Busan 48547, Korea (e-mails: pandu.sandy@gmail.com, haintmitu@yahoo.com, hakkyeong@pknu.ac.kr, kimdh2599@pukyong.ac.kr, kimsb@pknu.ac.kr).
* Corresponding author.

bcome to zero, backstepping control method based on Lyapunov stability is introduced. Finally, the effectiveness of the proposed algorithms and controller are verified by experiments. The experimental results show that the AGV successfully reaches the goal point with an acceptable small error.

2. SYSTEM DESCRIPTION

The AGV used in this paper is shown in Fig. 1. The dimension of the AGV is 100 cm x 60 cm x 80 cm. This system uses a differential drive wheeled configuration. Two driving wheels are mounted on the left and right sides of AGV, and are driven by two BLDC motors. Two passive castor wheels are installed in front and back sides of AGV to support the AGV. The laser navigation system NAV-200 used for positioning sensor with an accuracy ± 25 mm is mounted on the top of AGV. Industrial PC as the main controller is placed inside the AGV platform and touch screen monitor as the input and display is located on back side of AGV. The batteries for power supply are mounted in the middle of AGV.

The electrical design schematics diagram of AGV is shown in Fig. 2. AGV uses 2 encoders and one laser scanner LMS151 for SLAM. LMS151 is also used for both landmark detection and obstacle detection. A laser navigation system NAV200 is used for measuring the position in absolute coordinate.

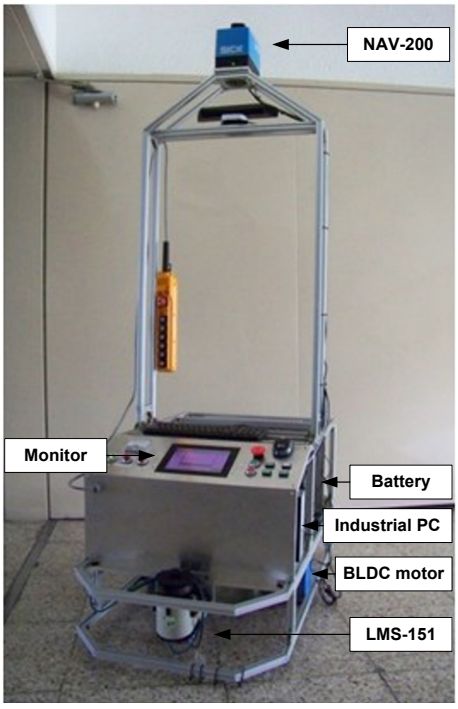


Fig. 1. The mechanical design of AGV.

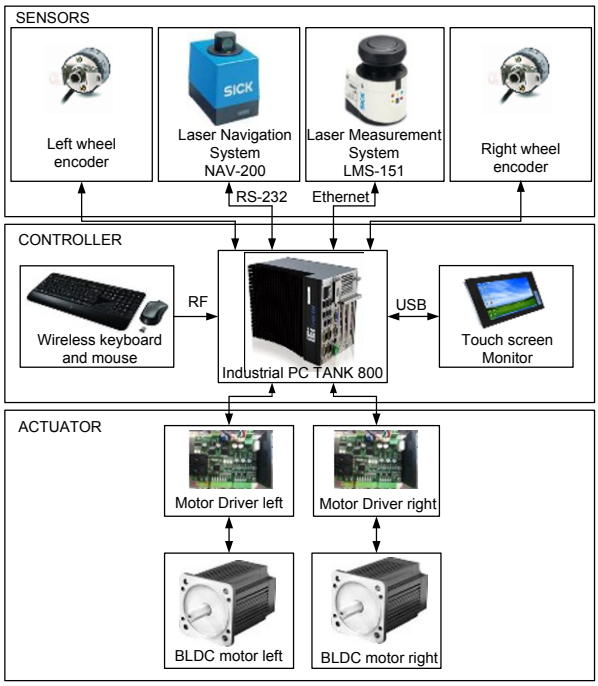


Fig. 2. Electrical design schematics diagram.

3. KINEMATICS MODELING

Fig. 3 shows the system modeling of the differential drive AGV system.

The kinematic equation of nonholonomic differential drive type of AGV system shown in Fig. 3 can be expressed as follows:

$$\mathbf{x}_v = \int \dot{\mathbf{x}}_v dt, \tag{1}$$

$$\dot{\mathbf{x}}_v = \begin{bmatrix} \dot{X}_A \\ \dot{Y}_A \\ \dot{\theta}_A \end{bmatrix} = \begin{bmatrix} \cos\theta_A & 0 \\ \sin\theta_A & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_A \\ \omega_A \end{bmatrix}, \tag{2}$$

$$\begin{bmatrix} V_A \\ \omega_A \end{bmatrix} = \frac{r}{2} \begin{bmatrix} 1 & 1 \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix}, \tag{3}$$

or in discrete type

$$\mathbf{x}_{vk} = \mathbf{x}_{vk-1} + \Delta\mathbf{x}_v, \tag{4}$$

$$\Delta\mathbf{x}_v = \begin{bmatrix} \Delta X_A \\ \Delta Y_A \\ \Delta\theta_A \end{bmatrix} = \begin{bmatrix} \cos(\theta_A + \Delta\theta) & 0 \\ \sin(\theta_A + \Delta\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta s \\ \Delta\theta \end{bmatrix}, \tag{5}$$

$$\begin{bmatrix} \Delta s \\ \Delta\theta \end{bmatrix} = \frac{r}{2} \begin{bmatrix} 1 & 1 \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \begin{bmatrix} \Delta\phi_r \\ \Delta\phi_l \end{bmatrix}, \tag{6}$$

where $\mathbf{x}_v$ is the posture vector of AGV, (X_A, Y_A) is AGV position in global coordinates, θ_A is the AGV orientation which is taken counterclockwise from the X axis, V_A is the linear velocity and ω_A is the angular velocity, r is the

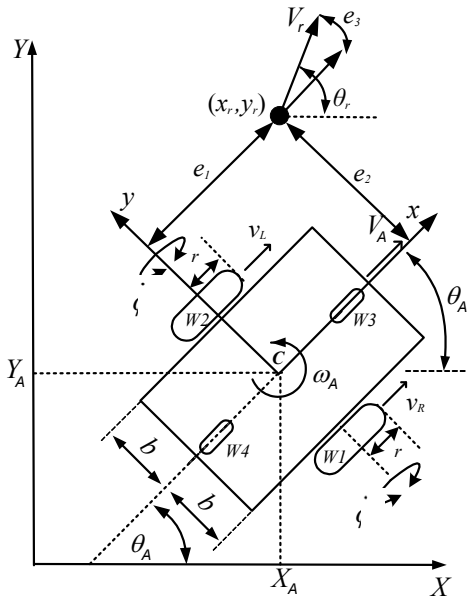


Fig. 3. System modeling.

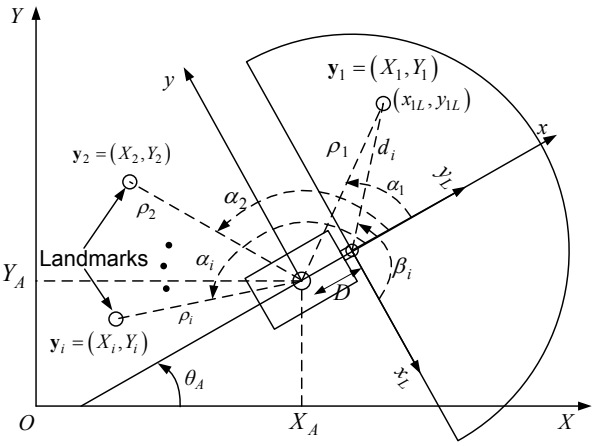


Fig. 4. Landmark detection.

wheel radius, b is the distance between the wheels and center of AGV, $\dot{\phi}_R$ and $\dot{\phi}_L$ are the right and left wheel angular velocities, Δs is the linear displacement of AGV, $\Delta\theta$ is the change of rotational angle of AGV, and $\Delta\phi_R$ and $\Delta\phi_L$ represent the change of right and left wheel rotation angle, respectively.

4. POSITIONING

In this paper, positioning of AGV can be obtained using SLAM algorithm based on EKF since the system is nonlinear. To do this, the following are done. Firstly, the positions of the landmarks are obtained using laser scanner LMS-151. Secondly, the position of the AGV is predicted using EKF prediction step based on encoder data. Thirdly, the positions of AGV and landmarks are updated using EKF update step based on landmark positions.

4.1. Landmark detection

In this paper, spike landmark extraction is used to extract the features that can be easily re-observed and distinguished from the environment. This method uses extrema to get landmarks by finding values in the range of current laser beam that differs by more than a certain amount such as 0.5 meters from previous laser beam. The landmarks detected using spike algorithm are shown in Fig. 4.

In this paper, laser scanner LMS-151 is used to detect landmarks and obstacles. The resolution of laser scanner is 0.5° and the scanning frequency is 25 Hz. The detected object can be expressed either with polar coordinate as (d_i,β_i), where d_i is the distance between sensor and object, and β_i is the scanning angle or with Cartesian coordinate in local coordinate frame of sensor as follows:

$$(x_{iL}, y_{iL}) = (d_i \cos\beta_i, d_i \sin\beta_i) \text{ for } i = 1,2,\ldots,n. \quad (7)$$

Using coordinate transformation, the position of landmark y_i in the global coordinate is obtained as:

$$\begin{bmatrix} X_{i,k} \\ Y_{i,k} \end{bmatrix} = \begin{bmatrix} X_{A,k} \\ Y_{A,k} \end{bmatrix} + \begin{bmatrix} \sin\theta_A & \cos\theta_A \\ -\cos\theta_A & \sin\theta_A \end{bmatrix} \begin{bmatrix} x_{iL} \\ y_{iL} \end{bmatrix} + D \begin{bmatrix} \cos\theta_A \\ \sin\theta_A \end{bmatrix}, \quad (8)$$

where (X_A, Y_A) is the position of AGV in global coordinate, θ_A is the orientation of AGV from X axis and D is the distance between the center of the robot and the sensor.

4.2. Prediction

The estimated position obtained from prediction step $\hat{\mathbf{x}}_{k|k-1}$ and the covariance matrix obtained from prediction step $\mathbf{P}_{k|k-1}$ at current sampling time k based on encoder data are written as follows:

$$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{F}_x^T f(\hat{\mathbf{x}}_{v,k-1}, \mathbf{u}_{k-1}), \quad (9)$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}_k \mathbf{P}_{k-1|k-1} \mathbf{A}_k^T + \mathbf{W}_k \mathbf{Q}_{k-1} \mathbf{W}_k^T, \quad (10)$$

where $\mathbf{x} = \begin{bmatrix} \mathbf{x}_v & \mathbf{y}_1 & \cdots & \mathbf{y}_n \end{bmatrix}^T$ is the state vector that consists of AGV posture vector $\mathbf{x}_v = \begin{bmatrix} X_A & Y_A & \theta_A \end{bmatrix}^T$ and the landmark position vector $\mathbf{y}_i = \begin{bmatrix} X_i & Y_i \end{bmatrix}^T$, $\hat{\mathbf{x}}_{k-1|k-1}$ is the estimated position obtained from the previous update, $\mathbf{F}_x = \begin{bmatrix} \mathbf{I}^{(3)} & \mathbf{O}^{(3)\times(2n)} \end{bmatrix}$ is used to make sure that only the AGV estimation position is updated, $f(\hat{\mathbf{x}}_{v,k-1}, \mathbf{u}_{k-1})$is the mathematic model of AGV, $\hat{\mathbf{x}}_{v,k-1}$ is the AGV estimated posture vector obtained from the previous update, $\mathbf{u}_{k-1}$ is the control input, $\mathbf{A}_k$ is the Jacobian of prediction model, $\mathbf{P}_{k-1|k-1}$is the covariance matrix obtained from the previous update, $\mathbf{W}_k$ is the process noise, $\mathbf{Q}_{k-1}$ is the process noise covariance.

This prediction is based on the input $\mathbf{u}_{k-1}$ that consists of changes of right encoder $\Delta\phi_r$ and left encoder $\Delta\phi_l$ respectively. AGV mathematic modeling in discrete type in (5) and (6) can be reduced as follows:

$$f(\hat{\mathbf{x}}_{v,k-1},\mathbf{u}_{k-1}) \equiv \Delta\mathbf{x}_v = \begin{bmatrix} \cos(\hat{\theta}_{A,k-1}+\Delta\theta) & 0 \\ \sin(\hat{\theta}_{A,k-1}+\Delta\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta s \\ \Delta\theta \end{bmatrix}, \quad (11)$$

$$\begin{bmatrix} \Delta s \\ \Delta\theta \end{bmatrix} = \frac{r}{2} \begin{bmatrix} 1 & 1 \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \begin{bmatrix} \Delta\phi_r \\ \Delta\phi_l \end{bmatrix},$$

$$\mathbf{u}_{k-1} = \begin{bmatrix} \Delta\phi_r & \Delta\phi_l \end{bmatrix}, \quad (12)$$

where b is the distance between the left and right wheels, Δs is the linear displacement of AGV, and $\Delta\theta$ is the rotational angle of AGV.

The covariance matrix $\mathbf{P}_{k|k-1}$ and Jacobian of prediction model $\mathbf{A}_k$ are defined as:

$$\mathbf{P}_{k|k-1} = \begin{bmatrix} P_{xx} & P_{xy_1} & \cdots & P_{xy_n} \\ P_{y_1x} & P_{y_1y_1} & \cdots & P_{y_1y_n} \\ \vdots & \vdots & \ddots & \\ P_{y_1x} & P_{y_ny_1} & & P_{y_ny_n} \end{bmatrix}, \quad (13)$$

$$\mathbf{A}_k = \mathbf{I}^{(3+2n)} + \mathbf{F}_x^T \mathbf{A}_{x,k} \mathbf{F}_x, \quad (14)$$

where

$$\mathbf{A}_{x,k} = \frac{\partial f}{\partial \mathbf{x}_v} = \begin{bmatrix} 0 & 0 & -\Delta s\sin(\theta_A+\Delta\theta/2) \\ 0 & 0 & \Delta s\cos(\theta_A+\Delta\theta/2) \\ 0 & 0 & 0 \end{bmatrix}. \quad (15)$$

Equation (14) is used to make sure that only the covariance matrix of AGV is updated.

The process noise $\mathbf{W}_k$can be calculated as:

$$\mathbf{W}_k = \mathbf{F}_x^T \mathbf{W}_{u,k}, \quad (16)$$

where process noise for AGV $\mathbf{W}_{u,k}$ and its covariance $\mathbf{Q}_{k-1}$ with error constants k_r and k_l are defined as:

$$W_{u,k} = \frac{\partial f}{\partial u} = \frac{1}{2} \begin{bmatrix} \cos(\theta_k) - \frac{\Delta s}{b}\sin(\theta_k) & \cos(\theta_k) + \frac{\Delta s}{b}\sin(\theta_k) \\ \sin(\theta_k) + \frac{\Delta s}{b}\cos(\theta_k) & \sin(\theta_k) - \frac{\Delta s}{b}\cos(\theta_k) \\ \frac{2}{b} & -\frac{2}{b} \end{bmatrix} \quad (17)$$

with

$$\theta_k = \theta_A + \frac{\Delta\theta}{2}, \ \mathbf{Q}_{k-1} = \begin{bmatrix} k_r|\Delta\phi_r| & 0 \\ 0 & k_l|\Delta\phi_l| \end{bmatrix}. \quad (18)$$

4.3. Update

The estimated position obtained from update step $\hat{\mathbf{x}}_{k|k}$ and the covariance matrix obtained from update step $\mathbf{P}_{k|k}$ at sampling time k based on landmarks data are written as follows:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k v_k \text{ for } \mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T \mathbf{S}_k^{-1}, \quad (19)$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1}, \quad (20)$$

where $\mathbf{K}_k$ is Kalman gain, $\mathbf{v}_k$ is the innovation as difference between landmark positions obtained from measurement and prediction. If the number of landmark is $i = 1$, 2, ..., n, the innovation of each landmark $\mathbf{v}_{i,k}$ is calculated as follows:

$$v_{i,k} = z_{i,k} - h(\hat{x}_{v,k|k-1}, \hat{y}_{i,k|k-1}), \quad (21)$$

$$h(\hat{\mathbf{x}}_{v,k|k-1}, \hat{\mathbf{y}}_{i,k|k-1}) = \begin{bmatrix} \hat{\rho}_{i,k|k-1} \\ \hat{\alpha}_{i,k|k-1} \end{bmatrix} = \begin{bmatrix} \sqrt{(\hat{X}_{i,k|k-1} - \hat{X}_{A,k|k-1})^2 + (\hat{Y}_{i,k|k-1} - \hat{Y}_{A,k|k-1})^2} \\ \tan^{-1}(\frac{\hat{Y}_{i,k|k-1} - \hat{Y}_{A,k|k-1}}{\hat{X}_{i,k|k-1} - \hat{X}_{A,k|k-1}}) - \hat{\theta}_{A,k|k-1} \end{bmatrix}, \quad (22)$$

$$z_i = \begin{bmatrix} \hat{\rho}_{i,k} \\ \hat{\alpha}_{i,k} \end{bmatrix} = \begin{bmatrix} \sqrt{(X_{i,k} - \hat{X}_{A,k|k-1})^2 + (Y_{i,k} - \hat{Y}_{A,k|k-1})^2} \\ \tan^{-1}(\frac{Y_{i,k} - \hat{Y}_{A,k|k-1}}{X_{i,k} - \hat{X}_{A,k|k-1}}) - \hat{\theta}_{A,k|k-1} \end{bmatrix}, \quad (23)$$

where range $\hat{\rho}_i$ is the estimated distance between AGV and current positions of landmarks, and $\hat{\alpha}_i$ is an estimated angle between AGV and landmark current position, $(\hat{X}_{A,k|k-1}, \hat{Y}_{A,k|k-1}, \hat{\theta}_{A,k|k-1})$ is the AGV posture obtained from prediction step, $(\hat{X}_{i,k|k-1}, \hat{Y}_{i,k|k-1})$is the landmark position obtained from prediction step. $\mathbf{z}_i$ is the measurement value vector of landmark obtained using landmark position $\mathbf{y}_i$ in global coordinate from (8).

The innovation covariance $\mathbf{S}_k$ can be defined as:

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}, \quad (24)$$

$$\mathbf{R} = \begin{bmatrix} \sigma_{\rho\rho} & 0 \\ 0 & \sigma_{\alpha\alpha} \end{bmatrix}, \quad (25)$$

where $\mathbf{R}$ is the noise covariance of sensor with measurement accuracy $\sigma_{\rho\rho}$ and $\sigma_{\alpha\alpha}$ of the sensor.

The detected landmarks are then associated with the previous known landmarks based on Mahalanobis distance (χ_i) that $\chi_i^2 = \mathbf{v}_i^T\mathbf{S}_i^{-1}\mathbf{v}_i$. The detected landmark belongs to the previous known landmark if $\chi_i^2 < \gamma$ where γ is threshold value [16].

Jacobian of measurement model $\mathbf{H}_k$ is defined as:

$$\mathbf{H}_k = \begin{bmatrix} \mathbf{H}_x & \mathbf{H}_y \end{bmatrix} \begin{bmatrix} \mathbf{F}_x \\ \mathbf{F}_y \end{bmatrix}, \quad (26)$$

where

$$\mathbf{H}_x = \frac{\partial h}{\partial \mathbf{x}_v} = \begin{bmatrix} -\frac{X_i - X_A}{\rho_i} & -\frac{Y_i - Y_A}{\rho_i} & 0 \\ \frac{Y_i - Y_A}{\rho_i^2} & -\frac{X_i - X_A}{\rho_i^2} & -1 \end{bmatrix}, \quad (27)$$

$$\mathbf{H}_y = \frac{\partial h}{\partial \mathbf{y}_i} = \begin{bmatrix} \frac{X_i - X_A}{\rho_i} & \frac{Y_i - Y_A}{\rho_i} \\ -\frac{Y_i - Y_A}{\rho_i^2} & \frac{X_i - X_A}{\rho_i^2} \end{bmatrix}, \tag{28}$$

$$\mathbf{F}_y = \begin{bmatrix} \mathbf{I}^{(2)\times(3)} & \mathbf{O}^{(2)\times(2(i-1))} & \\ & \mathbf{I}^{(2)\times(2)} & \mathbf{O}^{(2)\times(2(n-i+1))} \end{bmatrix},$$

where $\mathbf{H}_x$ is a Jacobian related with AGV position and $\mathbf{H}_y$ is a Jacobian related with landmark position.

If new landmarks are detected, the new landmarks are included in the state vector $\hat{\mathbf{x}}_{k|k}$ as follows:

$$\hat{\mathbf{x}}_{k|k} = \begin{bmatrix} \hat{\mathbf{x}}_{k|k-1} \\ y(\hat{\mathbf{x}}_{v,k|k-1}, \mathbf{z}) \end{bmatrix}, \tag{29}$$

where

$$y(\mathbf{x}_v, \mathbf{z}) = \begin{bmatrix} X_A + \rho\cos(\alpha + \theta_A) \\ Y_A + \rho\sin(\alpha + \theta_A) \end{bmatrix}. \tag{30}$$

The covariance matrix $\mathbf{P}_{k|k}$ can be obtained as follows:

$$\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{P}_{k|k-1} & \mathbf{P}^T_{x(x\sim y_n)}\mathbf{Y}_x^T \\ \mathbf{Y}_x\mathbf{P}_{x(x\sim y_n)} & \mathbf{Y}_x\mathbf{P}_{xx}\mathbf{Y}_x^T + \mathbf{Y}_z\mathbf{R}\mathbf{Y}_z^T \end{bmatrix}, \tag{31}$$

where

$$\mathbf{P}_{x(x\sim y_n)} = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy_1} & \cdots & \mathbf{P}_{xy_n} \end{bmatrix}, \tag{32}$$

$$\mathbf{Y}_x = \frac{\partial y}{\partial \mathbf{x}_v} = \begin{bmatrix} 1 & 0 & -\rho\sin(\alpha + \theta_A) \\ 0 & 1 & \rho\cos(\alpha + \theta_A) \end{bmatrix}, \tag{33}$$

$$\mathbf{Y}_z = \frac{\partial y}{\partial \mathbf{z}} = \begin{bmatrix} \cos(\alpha + \theta_A) & -\rho\sin(\alpha + \theta_A) \\ \sin(\alpha + \theta_A) & \rho\cos(\alpha + \theta_A) \end{bmatrix}. \tag{34}$$

From the above EKF calculation, the estimated position of the AGV $\hat{\mathbf{x}}_v = \begin{bmatrix} X_A & Y_A & \theta_A \end{bmatrix}$ can be obtained. These values are used as the feedback values for tracking controller in the next section.

5. PATH PLANNING

5.1. Grid map representation

In this paper, a grid map technique is used to represent the environment. Fig. 5 illustrates how LMS-151 scans the environment with resolution of 0.5°laser signal. The initial value sign to each grids is ∞. When an obstacle is detected by LMS-151, the obstacle position is calculated by (8). Then, the node of this position is considered as an object in the grid map. The value of the grid with obstacle is 1. The value of empty grid is 0. By incorporating this node in the grid map, it will be considered in path planning or replanning algorithm process.

5.2. D* Lite algorithm

The pseudo code of D* Lite algorithm is shown in Fig. 6. D* Lite algorithm utilizes a heuristic and a priority queue to perform its search and to order its cost updates.

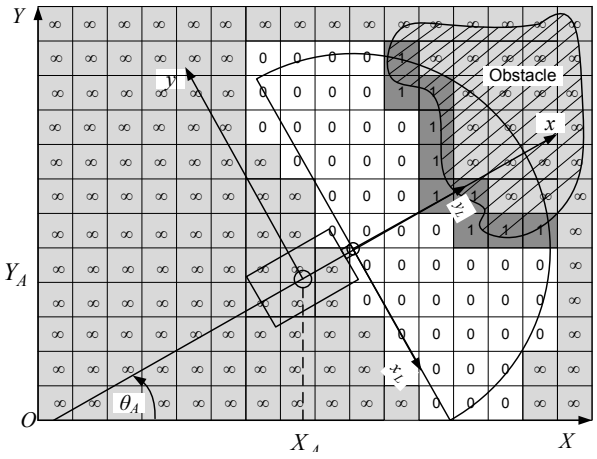


Fig. 5. Obstacle detection.

key(s)
01. return $[\min(g(s), rhs(s)) + h(s_{start}, s); \min(g(s), rhs(s))]$;

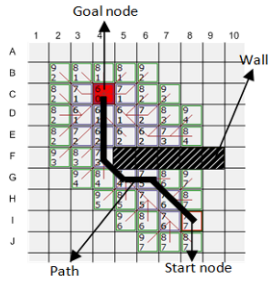
UpdateState(s)
02. if s was not visited before
03. $g(s) = \infty$;
04. if $(s \neq s_{goal})$ $rhs(s) = \min_{s' \in Succ(s)}(c(s, s') + g(s'))$
05. if $(s \in OPEN)$ remove s from $OPEN$;
06. if $(g(s) \neq rhs(s))$ insert s into $OPEN$ with key(s);

ComputeShortestPath()
07. while $(\min_{s \in OPEN}(key(s)) < key(s_{start})$ OR $rhs(s_{start}) \neq g(s_{start}))$
08. remove state s with the minimum key from $OPEN$;
09. if $(g(s) > rhs(s))$
10. $g(s) = rhs(s)$;
11. for all $s' \in Pred(s)$ UpdateState(s');
12. else
13. $g(s) = \infty$;
14. for all $s' \in Pred(s) \cup \{s\}$ UpdateState(s');

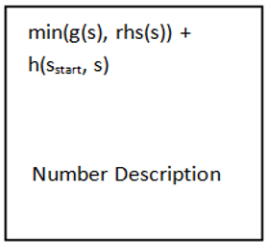
Main()
15. $g(s_{start}) = rhs(s_{start}) = \infty$; $g(s_{goal}) = \infty$;
16. $rhs(s_{goal}) = 0$; $OPEN = \emptyset$;
17. insert s_{goal} into OPEN with key(s_{goal});
18. forever
19. ComputeShortestPath();
20. Wait for changes in edge costs;
21. for all directed edges (u, v) with changed edge costs
22. Update the edge costs $c(u, v)$;
23. UpdateState(u);

Fig. 6. D* Lite algorithm [15].

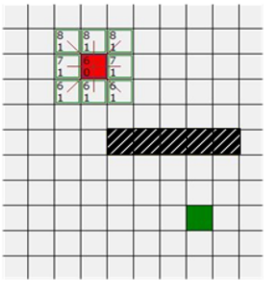
It calculates the path from the goal to the start position. In D* Lite algorithm, the heuristic value is the distance from the each grid to the goal. So if the state has 4 nodes away



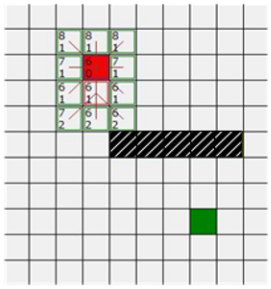
(a) D* Calculation.



(b) Number description of node.



(c) First step.



(d) Second step.

Fig. 7. Illustration of D* Lite algorithm path planning.

from the goal, the heuristic value is also 4.

The illustration of D* Lite algorithm is shown in Fig. 7. Fig. 7(a) shows the calculation result of D* Lite algorithm. Fig. 7(b) shows the number description of the node, whereas Fig. 7(c) and Fig. 7(d) show the first and second calculation steps of D* Lite algorithm, respectively. According to Koenig and Likhachev [15], D* Lite algorithm can be two times more efficient than A* algorithm in replanning a new path when the AGV encountered the obstacle. A* algorithm has to replan the path from the beginning whereas D* Lite algorithm does not replan the path as it already has information of the surrounding from the first search.

The output of D* Lite algorithm is a consecutive coordinate $\begin{bmatrix} s_1 & s_2 & \cdots & s_j \end{bmatrix}$ and $s_i = (s_{x,i}, s_{y,i})$ connecting start node $s_{start} = (X_{start}, Y_{start})$ and goal node $s_{goal} = (X_{goal}, Y_{goal})$. The distance between two coordinates depends on the size of each grid in the grid map.

The AGV moves at constant linear velocity. Therefore, the reference linear velocity $V_r(k)$ is predefined and has constant value. The reference value can be obtained from following calculations:

$$
\begin{aligned}
&\text{while } (X_r(k) \neq X_{goal} \text{ and } Y_r(k) \neq Y_{goal}) \\
&\quad \theta_r(k+1) = \text{atan2}\left((s_{y,i+1} - s_{y,i}), (s_{x,i+1} - s_{x,i})\right) \\
&\quad X_r(k+1) = X_r(k) + (V_r \Delta t)\cos(\theta_r(k)) \\
&\quad Y_r(k+1) = Y_r(k) + (V_r \Delta t)\sin(\theta_r(k)) \\
&\quad \omega_r(k+1) = (\theta_r(k+1) - \theta_r(k))/\Delta t \\
&\quad \text{if } (X_r(k+1) = s_{x,i+1}) \text{ and } (Y_r(k+1) = s_{y,i+1}) \\
&\qquad i = i+1 \\
&\quad \text{end} \\
&\text{end,}
\end{aligned}
\tag{35}
$$

where Δt is the sampling time.

6. TRACKING CONTROLLER

The purpose of this section is to design a trajectory tracking controller for AGV to track the reference position $(X_r(t), Y_r(t))$ and reference orientation $\theta_r(t)$ with reference linear velocity $V_r(t)$ and angular velocity $\omega_r(t)$ obtained from $D*$ Lite algorithm. As shown in Fig. 3, the tracking error vector and its time derivative are defined as follows:

$$
\begin{aligned}
e(t) &= \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta_A & \sin\theta_A & 0 \\ -\sin\theta_A & \cos\theta_A & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_r - X_A \\ Y_r - Y_A \\ \theta_r - \theta_A \end{bmatrix},
\end{aligned}
\tag{36}
$$

$$
\begin{aligned}
\dot{e}(t) &= \begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} \\
&= \begin{bmatrix} \cos e_3 & 0 \\ \sin e_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_r \\ \omega_r \end{bmatrix} + \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_A \\ \omega_A \end{bmatrix}.
\end{aligned}
\tag{37}
$$

To guarantee the stability of the system, Lyapunov function can be chosen as:

$$
V_0 = \frac{1}{2}e_1^2 + \frac{1}{2}e_2^2 + \frac{1}{k_2}(1 - \cos e_3) \text{ for } k_2 > 0, \tag{38}
$$

and its derivatives becomes

$$
\begin{aligned}
\dot{V}_0 &= e_1\dot{e}_1 + e_2\dot{e}_2 + \frac{1}{k_2}(\sin e_3)\dot{e}_3 \\
&= e_1(-V_A + V_r \cos e_3) + \frac{1}{k_2}(\sin e_3)(\omega_r - \omega_A + k_2 e_2 V_r).
\end{aligned}
\tag{39}
$$

To achieve $\dot{V}_0 \leq 0$, a control law vector $\mathbf{U}$ is chosen as follows:

$$
\mathbf{U} = \begin{bmatrix} V_A \\ \omega_A \end{bmatrix} = \begin{bmatrix} V_r \cos e_3 + k_1 e_1 \\ \omega_r + k_2 V_r e_2 + k_3 \sin e_3 \end{bmatrix}. \tag{40}
$$

The velocities of left and right wheels can be written from (3) as follows:

$$
\begin{bmatrix} \dot{\phi}_L \\ \dot{\phi}_R \end{bmatrix} = \begin{bmatrix} \frac{1}{r} & -\frac{b}{2r} \\ \frac{1}{r} & \frac{b}{2r} \end{bmatrix} \begin{bmatrix} V_A \\ \omega_A \end{bmatrix}. \tag{41}
$$

The above introduced total control loop for the AGV can be described as shown in Fig. 8.

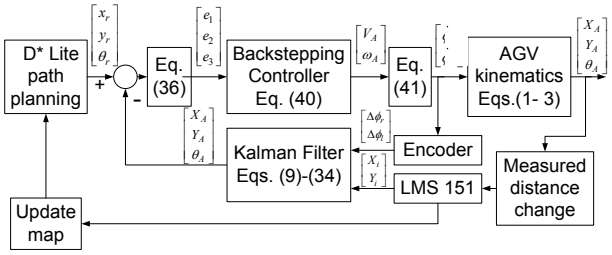


Fig. 8. Total control loop.

Table 1. Parameters and initial values.

Parameter	Values	Parameter	Values
r	0.09 m	b	0.3 m
k_1	0.7	$X_A(0)$	1.65 m
k_2	6	$Y_A(0)$	-2 m
k_3	0.5	$\theta_A(0)$	0 rad
$V_A(0)$	0 m/s	$\sigma_{\alpha\alpha}$	0.001
$\omega_A(0)$	0 rad/s	$\sigma_{\rho\rho}$	0.002
$\mathbf{P}_0$	$\mathbf{O}^{(3\times3)}$		

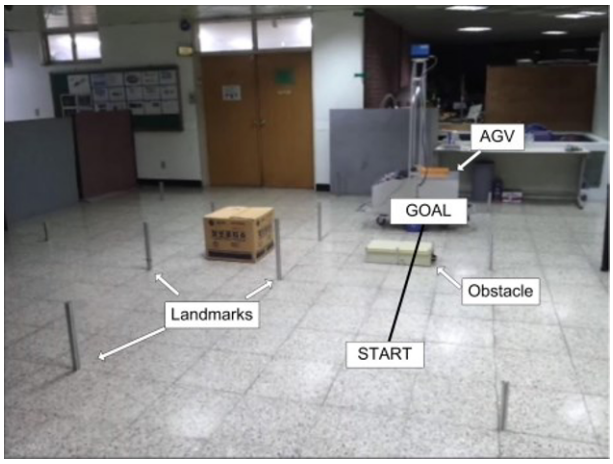


Fig. 9. Experimental environment.

7. EXPERIMENTAL RESULTS

The purpose of this paper is to generate a trajectory from start to goal position and to track the generated trajectory in partially known environment. In this section, the unknown environment is represented by obstacles that are not graphed in the given map.

To verify the effectiveness of the proposed algorithm, experiments are carried out. The parameters and initial values for experiment are shown in Table 1.

Fig. 9 shows the environment for the experiment of the proposed system. The environment consists of landmarks, known obstacles and unknown obstacles. Experimental environment in Fig. 9 can be represented in grid map as shown in Fig. 10. Grid size used in this paper is 40 × 40 cm.

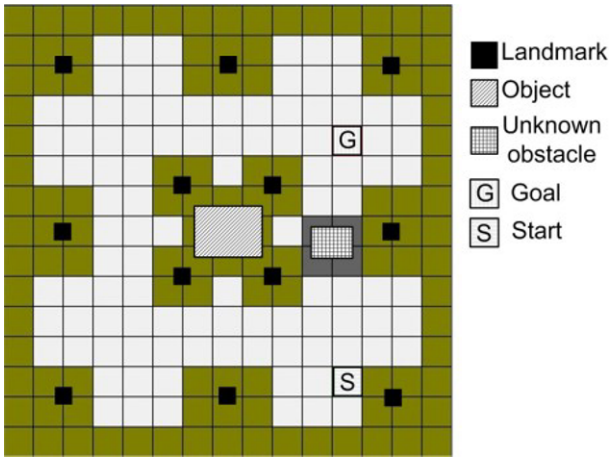


Fig. 10. Grid map of environment.

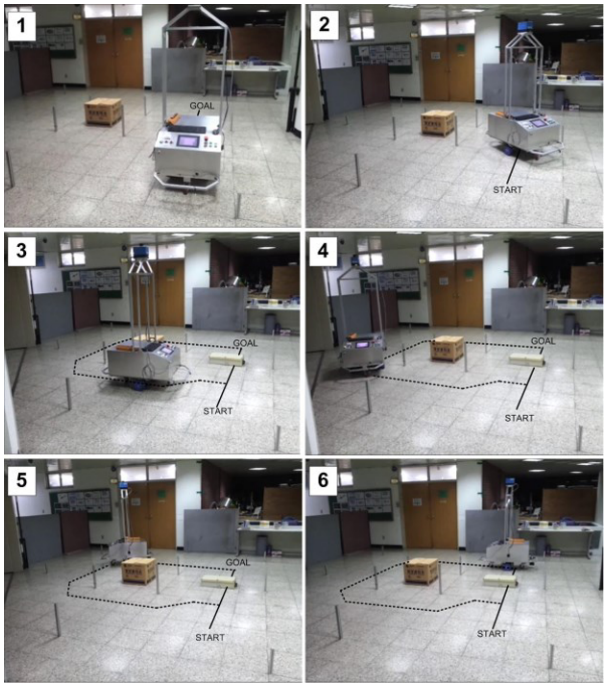
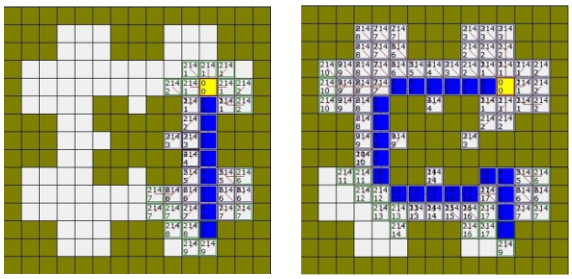


Fig. 11. Experimental result.

The positions of the AGV during the experiment are shown in Fig. 11. Fig. 11(1) shows an initial position of the AGV. In this position, since the obstacle in front of AGV is unknown, the AGV generates the trajectory as shown in Fig. 12(a). Fig. 11(2) shows the AGV detects an unknown obstacle and then replans the trajectory as shown in Fig. 12(b). Fig. 11(3-5) show that the AGV tracked the new generated trajectory and finally reaches the goal position of Fig. 11(6).

7.1. D* path planning and obstacle avoidance

The generated path obtained by D* Lite algorithm is shown in Fig. 12. As shown in Fig. 12(a), since at the start



(a) Path planning. (b) Path replanning.

Fig. 12. D* Lite path planning.

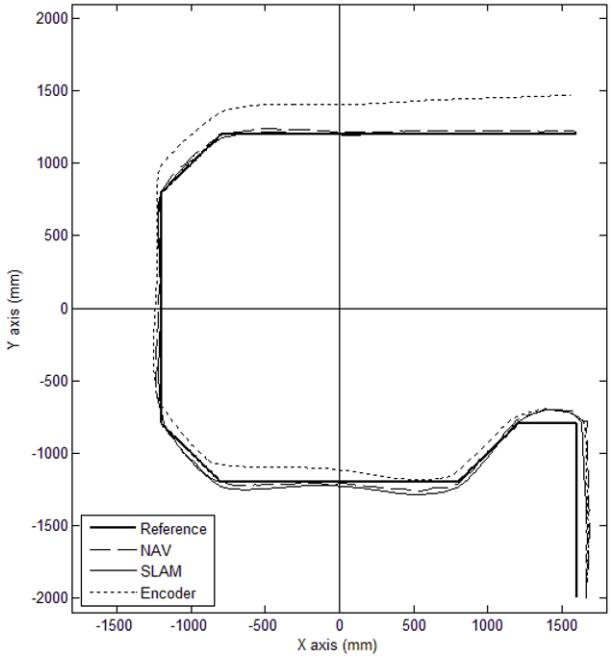


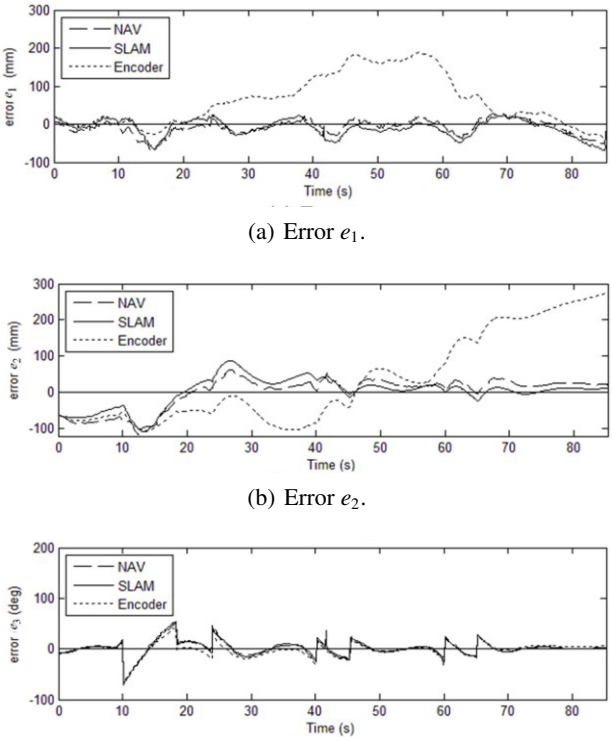
Fig. 13. Trajectory tracking.

state the robot does not know that there is an unknown obstacle in the surroundings, the robot plans the shortest path to the goal state that passes the unknown obstacle. After the path is built, the robot tracks the generated path until the robot meets the unknown obstacle and replans the path when it meets a n unknown obstacle.

7.2. Trajectory tracking

The trajectory tracking result is shown in Fig. 13. In this figure, different positioning method such as encoder, SLAM and NAV are compared. Fig. 13 shows that the controller successfully makes the AGV track the generated path with an acceptable small error. The position obtained by SLAM method is almost similar to absolute position measured by NAV.

The tracking error of the proposed controller is shown in Fig. 14. In this paper, the SLAM positioning method is



(a) Error e_1.

(b) Error e_2.

(c) Error e_3.

Fig. 14. Tracking error e_1, e_2, e_3 obtained from each sensors.

used as a feedback of trajectory tracking.This figure shows that the tracking errors e_1, e_2, e_3 obtained from SLAM are approaching to zero.

The error e_1 obtained from SLAM is increased at $t = 10$ s, $t = 20$ s, $t = 40$ s, $t = 60$ s since the direction of trajectory is changed. The error e_2 obtained from SLAM at t=0s is large since the initial position is different with the initial reference position. However, the controller successfully reduces the error. Therefore, the error e_2 is approaching zero after 20 s. The error e_3 obtained from SLAM is increased at $t = 10$ s, $t = 20$ s, $t = 40$ s, $t = 60$ s since the reference direction of the AGV is changed.

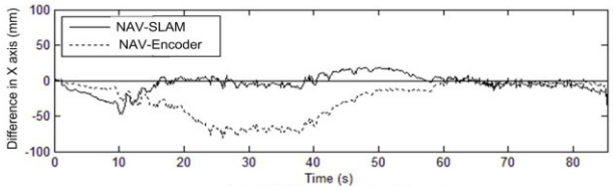
7.3. Positioning

To verify the effectiveness of SLAM positioning method, the positions obtained from encoders and SLAM are compared with the position obtained from NAV. NAV is a common positioning sensor used in industrial applications. The comparison of three different positioning methods is shown in Table 2 as follows:

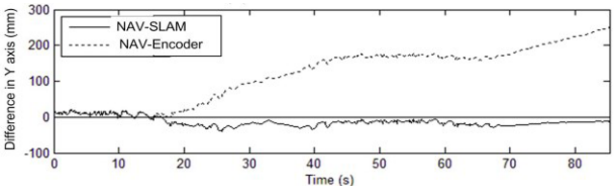
The experimental results regarding the difference among SLAM, encoder and NAV are shown in Fig. 15. Fig. 15 shows that the position difference between SLAM and NAV in X and Y axis is less than 20 mm. The angle difference between SLAM and NAV is less than 10°. On the other hand, the position difference between en-

Table 2. Comparison of NAV, encoder and SLAM.

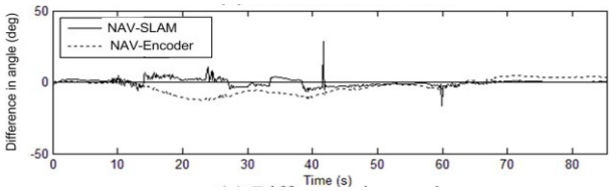
Items	NAV	Encoder	SLAM
Sensor	NAV-200	2 encoders	2 encoders and LMS-151
Positioning method	Triangulation method using mapped reflectors positioned around the environment	Based on the accumulation of wheel rotation	Combaining encoder positioning with landmark position using EKF
Positioning measurement error	Acorrding to datasheet $X = \pm(4 \sim 20)$ mm, $Y = \pm(4 \sim 20)$ mm, $\theta = \pm 0.1^\circ$	Compared to NAV $X = \pm 80$ mm, $Y = \pm 200$ mm, $\theta = \pm 20^\circ$	Compared to NAV $X = \pm 20$ mm, $Y = \pm 20$ mm, $\theta = \pm 10^\circ$



(a) Difference in X axis.



(b) Difference in Y axis.



(c) Difference in angle.

Fig. 15. Difference between SLAM, encoder and NAV positioning.

coder positioning and NAV is less than 80 mm in X axis, and more than 200mm in Y axis. Moreover, the angle difference between encoder positioning and NAV is less than 20°. The proposed SLAM algorithm obtains better positioning result compared to encoder positioning.

8. CONCLUSION

Positioning and obstacle avoidance of AGV in partially known environment was proposed. SLAM algorithm based on EKF was proposed to obtain the position of AGV. For path planning and obstacle avoidance, $D*$ algorithm was proposed based on a given map and laser scanner data. To guarantee that the tracking errors approach to zero, a backstepping control method based on Lyapunov stability was proposed. The effectiveness of the proposed algorithms and controller was verified using experiment. The experimental results showed that the AGV successfully reached the goal point when there was an unknown obstacle with an acceptable error.

REFERENCES

[1] P. T. Doan, T. T. Nguyen, V. T. Dinh, H. K. Kim, and S. B. Kim, "Path tracking control of automatic guided vehicle using camera sensor," *Proceedings of the 1st International Symposium on Automotive and Convergence Engineering*, pp. 20-26, 2011.

[2] C. Chen, B. Wang, and Q. T. Ye, "Application of automated guided vehicle (AGV) based on inductive guidance for newsprint rolls transportation system," *Journal of Donghua University*, vol. 21, no. 2, pp. 88-92, 2004.

[3] S. Y. Lee and H. W. Yang "Navigation of automated guided vehicles using magnet spot guidance method," *Journal of Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 425-436, June 2012. [click]

[4] S. C. Yuan and L. Yao, "Robust type-2 fuzzy control of an automatic guided vehicle for wall-following," *Proceedings of International Conf. of Soft Computing and Pattern Recognition,* pp. 172-177, 2009. [click]

[5] P. S. Pratama, B. T. Luan, T. T. Tran, H. K. Kim, and S. B. Kim, "Trajectory tracking algorithm for automatic guided vehicle based on adaptive backstepping control method," *AETA 2013: Recent Advances in Electrical Engineering and Related Sciences/ Lecture Notes in Electrical Engineering*, vol. 282, pp. 535-544, 2014. [click]

[6] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems EPFL*, pp. 2383-2388, 2002.

[7] D. H. Kim, "Escaping route method for a trap situation in local path planning," *International Journal of Control, Automation, and Systems*, vol. 7, no. 3, pp. 495-500, 2009. [click]

[8] K. Konolige, "A gradient method for realtime robot control," *Proceedings of International Conf. on Intelligent Robots and Systems*, pp. 639-646, 2000. [click]

[9] R. Simmons, "The curvature-velocity method for local obstacle avoidance," *Proceedings of International Conference on Robotics and Automation*, pp. 3375-3382, 1996. [click]

[10] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics and Automation*, vol. 4, no. 1, pp. 23-33, 1997. [click]

[11] P. S. Pratama, S. K. Jeong, S. S. Park, and S. B. Kim, "Moving object tracking and avoidance algorithm for differential driving AGV based on laser measurement technology," *International Journal of Science and Engineering*, vol. 4, no. 1, pp. 11-15, January 2013.

[12] Y. Dai and S. G. Lee, "Formation control of mobile robots with obstacle avoidance based on GOACM using onboard sensors," *International Journal of Control, Automation, and Systems*, vol. 12, no. 5, pp. 1077-1089, August 2014. [click]

[13] N. Hung, J. S. Im, S. K. Jeong, H. K. Kim, and S. B. Kim, "Design of a sliding mode controller for an automatic guided vehicle and its implementation," *International Journal of Control, Automation, and Systems*, vol. 8, no. 1, pp. 81-90, 2010.

[14] T. A. Tamba, B. H. Hong, and K. S. Hong, "A path following control of an unmanned autonomous forklift," *International Journal of Control, Automation, and Systems*, vol. 7, no. 1, pp. 113-122, March 2009. [click]

[15] Y. D. Setiawan, P. S. Pratama, S. K. Jeong, V. H, Duy, and S. B. Kim, "Experimental comparison of A* and D* lite path planning algorithms for differential drive automated guided vehicle," *AETA 2013: Recent Advances in Electrical Engineering and Related Sciences/ Lecture Notes in Electrical Engineering*, vol. 282, pp. 555-564, 2014. [click]

[16] N. M. Kwok, Q. P. Ha, and G. Fang, "Data association in bearing-only SLAM using a cost function-based approach," *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 4108-4113, 2007. [click]

Pandu Sandi Pratama was born in Indonesia on November 1, 1986. He received his B.S. degree in Electrical Engineering Dept. of Diponegoro University, Indonesia in 2011. He then received the M.S degree in the Interdisciplinary Program of Mechatronics Engineering Dept., Pukyong National University, Busan, Korea in 2013. He then received a Ph.D. degree in the Dept. of Mechanical Engineering, Pukyong National University, Busan, Korea in 2015. His research fields of interest are computer science, robotic and mobile robot.

Trong Hai Nguyen was born in Vietnam on February 1, 1975. He received his B.S. and M.S. degree in Dept. of Electronics and Telecommunication, Hochiminh City University of Technology, Vietnam in 1999 and 2003. He is currently a student in doctor degree course at Pukyong National University, Busan, Korea. His research fields of interest are nonlinear control, robust control, path planning algorithm, conveyor control.

Hak Kyeong Kim was born in Korea on November 11, 1958. He received his B.S. and M.S. degrees in Dept. of Mechanical Engineering from Pusan National University, Korea in 1983 and 1985. He received his Ph.D. degree from the Dept. of Mechatronics Engineering, Pukyong National University, Busan, Korea in February, 2002. His fields of interest are robust control, biomechanical control, mobile robot control, and image processing control.

Dae Hwan Kim was born in Korea on March, 1982. He received his B.S. degree in Electrical Engineering from Chosun University, Kwangju, Korea in 2008. He then received his M.S and Ph.D degrees in Mechanical engineering from the Pukyong National University, Busan, Korea, in 2009 and 2015, respectively. His fields of interests are robust control, combustion engineering control, and mobile robot control.

Sang Bong Kim was born in Korea on August 6, 1955. He received his B.S. and M.S. degrees from National Fisheries University of Busan, Korea, in 1978 and 1980. He received his Ph.D. degree from Tokyo Institute of Technology, Japan in 1988. After then, he is a Professor of the Dept. of Mechanical Engineering, Pukyong National University, Busan, Korea. His research has been on robust control, biomechanical control, and mobile robot control.