Localization of mobile robot by matching three-dimensional data of SOKUIKI sensor and aerial imagery

Kohei Ito and Keisuke Yokota and Akihisa Ohya

Abstract— **In recent years, the demand for autonomous mobile robots in outdoors environments has been increasing. In order to guide the robot, it is essential to get the current position. In this study, we propose a method to correct the self-position of the robot, which is estimated by odometry, aerial imagery, and the SOKUIKI sensor. First, we explore how to superimpose the reflection point cloud on top of the aerial imagery. Next, matching the edge data of aerial imagery and the upper edge of the building from the inclined reflection point cloud corrects the self-correction of the robot, by which the robot is able to run on the straight road for about 60m autonomously.**

I. INTRODUCTION

In recent years, the demand for robots to do troublesome or difficult work for humans has been increasing. In order to guide the robot, it is essential to get the current position. Odometry is most popular method for self localization outdoors or indoors, but error occurs because it cannot detect tire idling and side slip. The errors become large in proportion to the time; therefore, in outdoor environments large odometry errors occur because the road surface is not smooth. One way to correct self position estimated by odometory, is to match the shape of the surrounding map obtained by the SOKUIKI Sensor,[1] mounted on the robot and be able to move, allowing the SOKUIKI sensor to acquire the reflection point cloud data of the surrounding buildings. The robot generates the map by accumulating point cloud data of the virtual space; [2][3] because of this we have to move the robot once to generate a map and can not navigate the robot in unknown outdoor environments. In this study, we try to navigate the mobile autonomous robot on the specified path using aerial imagery and the SOKUIKI sensor. Previous experimentation attempted to negative the mobile robot using aerial imagery and an electronic map[4][5] ; these used the data from the SOKUIKI sensor, and can not applied to aerial imagery taken from the oblique. On other hand, we can match three dimensional sensor data to the aerial imagery.

Kohei Ito is with Department of Mechanical Engineering, Kanazawa Technical College , 2-270 Hisayasu Kanazawa,Ishikawa 9218601, Japan kouhei_ito@kanazawa-tc.ac.jp

Keiske Yokota is with Master's Program in Computer Science, University of Tsukuba, 1-1-1 Tennodai Tsukuba, Ibaraki 3058573, Japan yokota @ roboken.cs.tsukuba.ac.jp

Akihisa Ohya is with Division of Information Engineering, University of Tsukuba, 1-1-1 Tennodai Tsukuba, Ibaraki 3058573, Japan ohya @ cs.tsukuba.ac.jp

II. ROBOT

A. Kinematics of the robot

We used a Power Wheel Steering (PWS) robot developed in our laboratory [Fig.1, Fig.2, Table.I] which has two components: linear velocity $v(t)$ and angular velocity $\omega(t)$; these linearly depend on the angular velocity of the right and left wheels. Specifically, the radiuses of the right and left wheels R_r and R_l, and the axle track is T; allowing for the calculation of $v(t)$ and $\omega(t)$ by follow equation:

$$\begin{pmatrix} v(t) \\ \omega(t) \end{pmatrix} = \begin{pmatrix} \frac{R_r}{2} & \frac{R_l}{2} \\ \frac{R_r}{T} & -\frac{R_l}{T} \end{pmatrix} \begin{pmatrix} \omega_r(t) \\ \omega_l(t) \end{pmatrix} \tag{1}$$

The robot's position and orientation $(x(t),y(t),\theta(t))$ are determined using the initial conditions $(x(0),y(0),\theta(0))$ the following equations:

$$\begin{cases} x(t) = \int_0^t v(\tau)\cos\theta(\tau)d\tau + x(0) \\ y(t) = \int_0^t v(\tau)\sin\theta(\tau)d\tau + y(0) \\ \theta(t) = \int_0^t \omega(\tau)d\tau + \theta(0) \end{cases} \tag{2}$$

We estimate the relative self-position of the robot from the start point using Odometry, and correct the errors by which aerial imagery and SOKUIKI sensor data.

B. Configuration of the Robot

This robot was built with a single board SH2 controller and controlled using serial commands from a laptop computer; the controller calculates torque by comparing encoder feedback to angular velocity, and the robot wheel motor is controlled by a PWM signal.

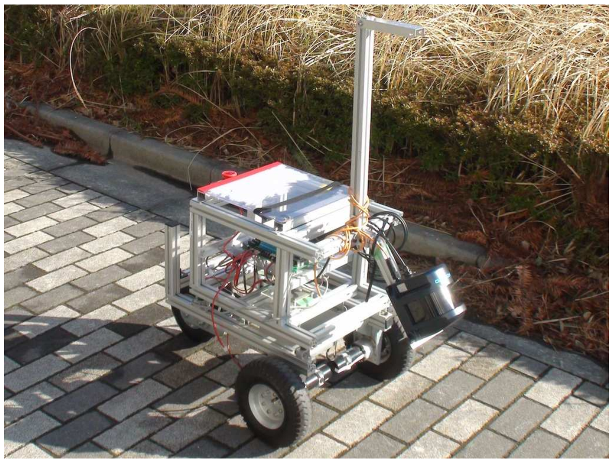

Fig. 1. PWS mobile robot

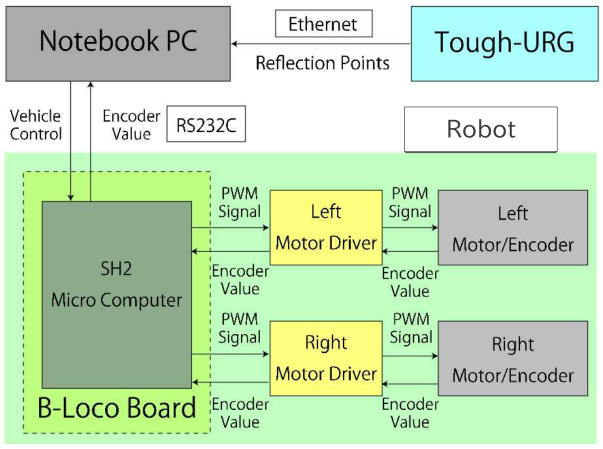

Fig. 2. System block Diagram

Robot control is achieved through original software developed in our laboratory; the robot can follow lines or circles using this software.

C. SOKUIKI Sensor

This sensor scans using a laser beam, and measures both distance and light receiving angle to the reflection points. This sensor is less influenced by the sun compared to cameras and supersonic wave sensors and has high range finding precision. We used the UXM-30LX-E(Tough-URG) developed by Hokuyo electricity, and aerial imagery instead of a map. Therefore, we have to measure from the ground to the upper edge of the building. We accomplished this by mounting the sensor 40cm above around level and angled upward at 45 degree angle on the front of the robot; we conjecture that the inclined sensor can measure more points than a sensor capturing the vertical axis.

TABLE I
SPECIFICATION OF THE ROBOT

Size	500mm×600mm×1100mm
Mass	about 22kg
Battery	Pb 12V × 2
Motor	60W × 2
Control board	B-LOCO
Contorol micro computer	SH7045F(SH2)
Softoware	YP-Spur

TABLE II
SPECIFICATION OF UXM-30LX-E (TOUGH-URG)

Interface	Ethernet
Power source	DC10〜30V
Light source	Semiconductor laser diode(λ=905nm)
Range	0.1〜30m
Max range	60m
Accuracy	≤3,000lx (0.1〜10m: ±50mm) ≤100,000lx (0.1〜30m: ±100mm)
Scan angle	190deg
Angular Resolution	約 0.25deg
Scan time	50ms
Weight	Approx. 800g（ without cable attachment）

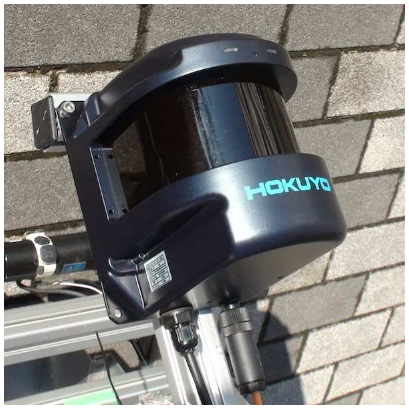

Fig. 3. UXM-30LX-E (Tough-URG)

III. SUPERIMPOSING THE AERIAL IMAGERY AND THREE DIMENSIONAL SOKUIKI SENSOR DATA

In this experiment, we want to see if we can match SOKUIKI sensor data with the robot movement and

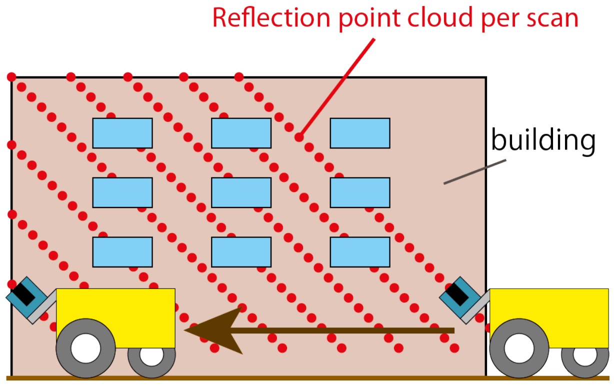

Fig. 4. How to get the point cloud of reflection

the aerial imagery.

A. Experiment conditions

We measure at a location meeting the following three conditions to lower Odometry error: the road surface has been fully paved, there are no steps, and the ground is not inclined. We measured at the gymnasium of the University of Tsukuba at midnight to capture only the wall, and ran the robot at 0.3m/s to accumulate the reflect point cloud data from the Tough-URG in three dimensional space.

B. Experiment

The aerial imagery was superimposed over the Canny edge data generated by OpenCV[6] to clear the edge in Fig.5. We can only infer the bottom edge of the building from the point cloud data on the third gymnasium , because the aerial imagery does not show the bottom edge of the building. In Fig.6, we superimposed three dimensional reflect point cloud data onto a horizontal plane of the aerial imagery edge data . Therefore, we can not correct the self-position using this superimposition method. As mentioned previously, the aerial imagery was not taken from directly above. We aimed to superimpose the inclined three dimensional reflect point cloud data imposed onto the upper and bottom edges of the building. Specifically, θ is the incline pitch angle and ϕ is the incline roll angle; we applied the following expressions to the reflect point(x_i,y_i,z_i) in three dimensional space, and (x_i,y_i,z_i) was converted into (x'_i,y'_i,z'_i); θ and ϕ were read manually from the aerial imagery as θ = -0.82 and ϕ = 0.36.

$$\begin{pmatrix} x'_i \\ y'_i \\ z'_i \end{pmatrix} = \begin{pmatrix} x_i + z_i \cos\theta \sin\phi \\ y_i + z_i \sin\theta \sin\phi \\ z_i \cos\phi \end{pmatrix} \quad (3)$$

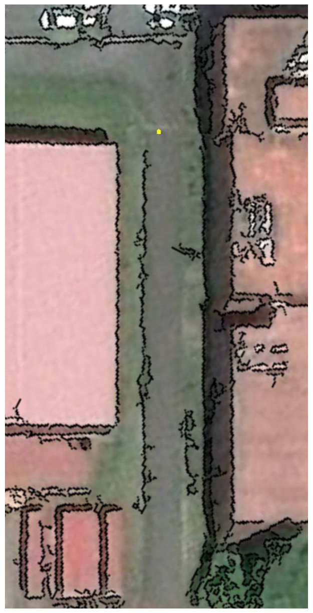

Fig. 5. Original Aerial imagery (it contains Canny edge data)

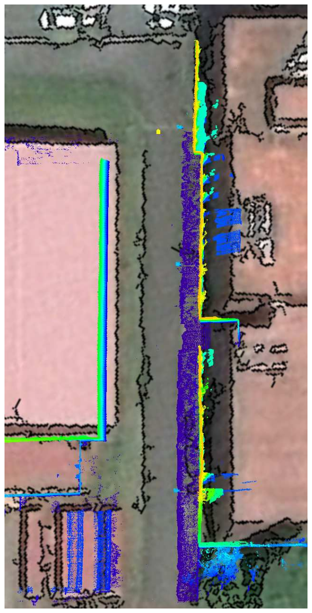

Fig. 6. Result of superposition of the reflection point cloud

Using this formula we can incline the reflect point cloud as shown in Fig.7. The data containing the highest point of every scan data was accumulated as shown in Fig.8.

IV. The Edge of the Aerial Imagery and Correcting the Error of the Self-Position by the Reflect Point Cloud Data

We implemented the algorithm to correct the error of the self-position by comparing the line from the edge data of the aerial imagery with the reflect point cloud data.

A. Implementation of the algorithm correcting the self-position

1) Selection of the upper edge of the building from the aerial imagery: We select the upper edge of the building from Fig.9, because the difference was clear and the edge is shown clearly by the Canny edge detector as shown in Fig.10 We selected the line l_{edge} drawn manually along the edge of Canny operation as the edge line to compare with the reflect point cloud data during the robot's movement.

2) Detection of the upper edge: We detected a straight line from the point group P, which was the point group accumulated at the highest point during every scan as shown in Fig.11. A chosen condition is that the point cloud data P' is not less than 1m from the straight line of l_{edge} as shown in Fig.12(1). Finally, we

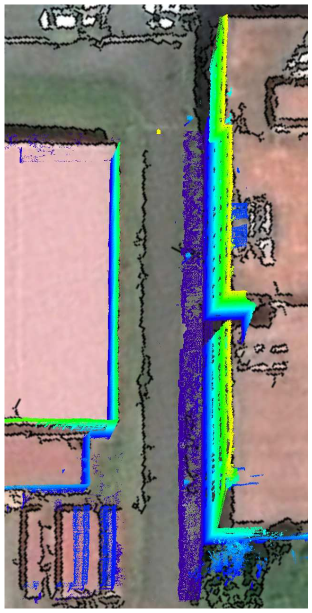

Fig. 7. Result of superposition of the reflection point cloud, which is tilted

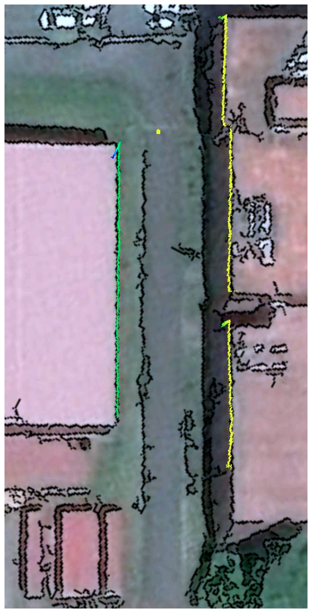

Fig. 8. Result of extracting the highest points for each scan data from from Fig.7

Fig. 9. Third Gymnasium, University of Tsukuba (DigitalGlobe, 2011)

calculated the straight line l_{LSM} from P' by the least-squares method as shown in Fig.12(2). The equation of line l_{edge} is $a_{edge}x+b_{edge}y+c_{edge}=0$, and the point cloud accumulated at the highest point is the P, thereby P is given by following equation:

$$P' = \left\{ (x,y) \mid (x,y) \in P \wedge \frac{|a_{edge}x+b_{edge}y+c_{edge}|}{\sqrt{a_{edge}^2+b_{edge}^2}} \leq 1.0 \right\} \tag{4}$$

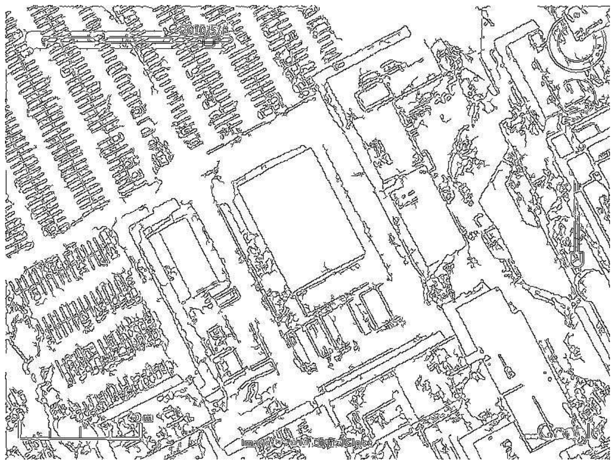

Fig. 10. Canny edege data of third Gymnasium

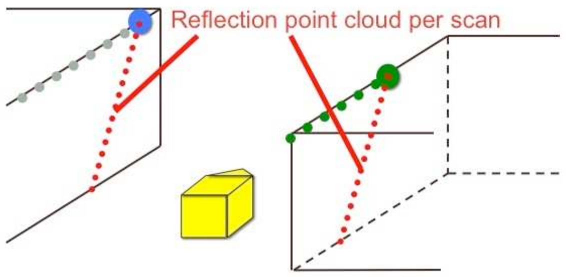

Fig. 11. How to extract the highest points of the reflection point cloud

The line l_{LSM} is calcurated from P' using the least-squares method. The number of point group $P'((x_i,y_i) \in P')$ is n, the equation of line l_{LSM} is $a_{LSM}x+b_{LSM}y+c_{LSM}=0$, where $a_{LSM}, b_{LSM}, c_{LSM}$ are given by the following equations:

$$\begin{cases} a_{LSM} &= n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i \\ b_{LSM} &= \left(\sum_{i=1}^{n} x_i\right)^2 - n\sum_{i=1}^{n} x_i^2 \\ c_{LSM} &= \sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i - \sum_{x=1}^{n} x_i y_i \sum_{x=1}^{n} x_i \end{cases} \tag{5}$$

3) Correction of self-position: We calculate the transformation matrix by which the straight line l_{edge} can be superimposed onto the straight line l_{LSM}. First, we find the point (x_c,y_c) which is the shortest point on the l_{LSM} from the oldest reflect point (x_0,y_0) of

P'(Fig.12(2)).

$$\begin{cases} x_c = \dfrac{b_{LSM}^2 x_0 - a_{LSM} b_{LSM} y_0 - c_{LSM} a_{LSM}}{a_{LSM}^2 + b_{LSM}^2} \\ y_c = \dfrac{-a_{LSM} b_{LSM} x_0 + a_{LSM}^2 y_0 - b_{LSM} c_{LSM}}{a_{LSM}^2 + b_{LSM}^2} \end{cases} \quad (6)$$

We translate the line l_{LSM} as if the point (x_c, y_c) is overlapped onto the line l_{edge}. Matrix A is the affine transformations matrix realizing this operation (Fig.12(3)).

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -\frac{a_{edge}}{b_{edge}} x_c - \frac{c_{edge}}{b_{edge}} - y_c \\ 0 & 0 & 1 \end{pmatrix} \quad (7)$$

Next, we calculate the affine transformations matrix B which rotate the line l_{LSM} to the l_{edge} by angle theta around the point (x'_c, y'_c) translated from (x_c, y_c) by the matrix A(Fig.12(4)).

$$B = \begin{pmatrix} \cos\theta & -\sin\theta & x_0(1-\cos\theta) + y_0 \sin\theta \\ \sin\theta & \cos\theta & -x_0 \sin\theta + y_0(1-\cos\theta) \\ 0 & 0 & 1 \end{pmatrix} \quad (8)$$

After the correction of the self-position, if the robot move more than 1m away from the path, the robot return to detecting the upper edge of the building.

B. Moving Experiment

We tested using both the above method and Odometry on the straight road as shown in Fig.5; Fig.13 shows the accumulated reflect point cloud data by both of them, once the robot finished traveling 60m. Using only Odometry, the robot moved gradually to the right, and collided with the curb after 45m. However, using the algorithm to correct the self-position, the robot traveled in the center of road with a few corrections of position and was able to move 60m. In Fig.13, the red points are the reflect point cloud data and the green line is the upper edge line of building. Using only Odometry; the reflect point cloud gradually deviated from the edge line, because the robot could not correct the angular error of Odometry; but when using the algorithm to correct the self-position, the reflect point cloud was almost overlapped by the edge line and we confirmed corrections in the robot's self-positioning.

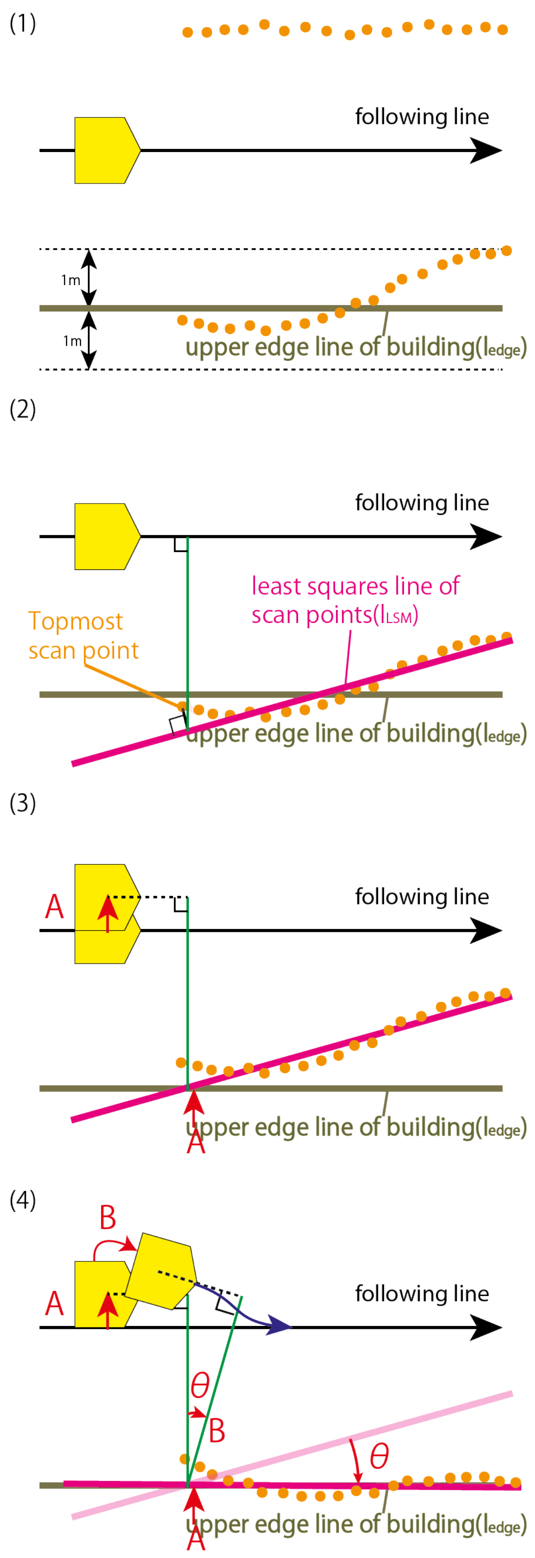
(1)
following line
1m
1m
upper edge line of building(l_edge)
(2)
following line
least squares line of scan points(l_LSM)
Topmost scan point
upper edge line of building(l_edge)
(3)
A
following line
upper edge line of building(l_edge)
A
(4)
B
A
following line
θ
B
θ
upper edge line of building(l_edge)
A

Fig. 12. How to correct the self-position of the robot

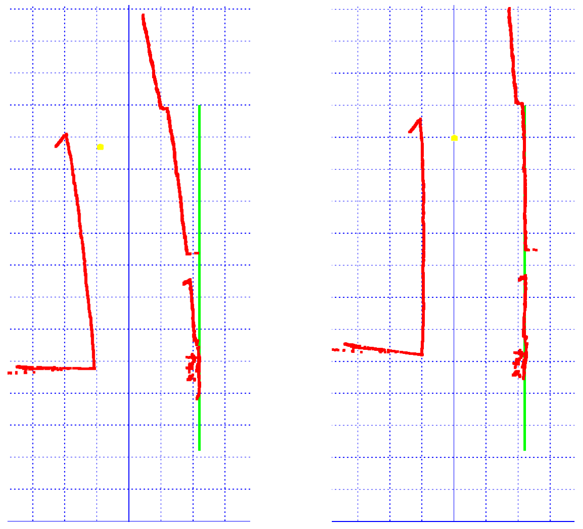
Fig. 13. Result of accumulating the reflection point cloud (left: self-localization by odometry, right: self-position correction)

V. Discussion

The robot turned left after the first 3m of travel, and we thought that the robot detected the wrong line because the influence of error was large initially when the robot didn't accumulate an adequate reflect point cloud. After which the robot accumulated an appropriate reflect point cloud, and the accuracy of the least-squares method increased. This problem is solved by delaying the start time of detecting the line. But, the Odometry errors increase when the robot turn, and we must solve this problem using a different method.

VI. CONCLUSIONS AND FUTURE WORKS

A. Conclusions

We proposed a method by which the robot can correct its position in an unknown outdoor environment. We tested the PWS mobile robot moving autonomously on the University of Tsukuba campus. We moved the robot 60m autonomously without colliding with the curb by matching the upper edge of the building to the aerial imagery data.

B. Future Works

In this experiment, we assumed that the robot travel in a straight line. But, the robot usually doesn't move in a straight line;therefore our future work is to study methods to correct the Odometry error of turning at the building's corner so that the robot can move around the building autonomously. We manually input the angle parameter at which the reflect point cloud was inclined from the SOKUIKI sensor; but we actually need to study methods to automatically retrieve this parameter from the aerial imagery, because these parameters are always different according to the angle of the picture. We also accumulated the highest point per scan to detect the upper edge of the building from the reflect point cloud, but we didn't explicitly detect the corners. If we can detect the corners of a building, we can expect increased precision in the self localization, because the object matching is improved.

References

[1] Sebastian Thrun, Wolfram Burgard, and Dieter Fox :“ Probabilistic Robotics ” , *The MIT Press*, 2005.

[2] Jyunichi Iwai, Satoshi Muramasu , Tetsuo Tomizawa, Takashi Suehiro, Shunsuke Kudoh : ”Environmental map generation based on free space model considering the reflection characteristics of objects.”, *Robomec2011*, 2A1-O03, 2011.

[3] Yosuke Furukawa, Ryosuke Ikezawa, Yusuke Osaki, Hiroshi An : ”Autonomous running method for obtaining environment map information by using a wheeled mobile robot”, *Robomec2011*, 2A1-O16, 2011.

[4] Taketoshi Mori, Takehiro Sato, Aiko Kuroda, Makoto Kurihara, Masayuki Tanaka, Masamichi Shimosaka, Rui Fukui, Tomomasa Sato, Hiroshi Noguchi : ”Outdoor Map Construction Based on Aerial Photography and Electrical Map Using Multi-Plane Laser Scan Data”, *SI2010*, 2O3-1, 2010.

[5] Fumiaki Nakazawa , Yoshinobu Ando, Takashi Yoshimi, Makoto Mizukawa : ”Research on landmark map generation for outdoor autonomous mobile robot using map software and self-position correction using generated map”, *SI2010*, 2O3-5, 2010.

[6] OpenCV, http://opencv.jp/, 2012.