Enhancing 3D range image measurement density via dynamic Papoulis–Gerchberg algorithm

Elvan Kuzucu^{1*}, Dilan Öztürk^{2*}, Mustafa Gül^{2*}, Bengisu Özbay^{3*},
A Mansur Arisoy^{4*}, H Onur Sirin⁵ and Ismail Uyanik⁶

Transactions of the Institute of
Measurement and Control
2018, Vol. 40(16) 4407–4420
© The Author(s) 2018
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/0142331218759899
journals.sagepub.com/home/tim

Abstract

As one of the most popular range detection methods, lidar is commonly used in various robotic applications. Although most robotic platforms easily adopt 2D lidar for range sensing, 3D lidar is rarely used in mobile robots, owing to its high cost. Some methods reported in the literature obtain 3D range information by rotating a single 2D lidar device. However, for most of these methods, there is a trade-off between 3D scan frequency and measurement density. Existing methods discussed in the literature for increasing the measurement density in high-frequency lidar have high time complexity and require certain conditions on data distribution. In a previous work, we showed the usability of an image super-resolution method, the Papoulis–Gerchberg (P–G) algorithm, on range data represented in the form of a greyscale image. However, the low convergence rate of the original P–G algorithm impedes its use for online applications. In this study, we advanced the P–G algorithm to drastically reduce the convergence time and improve performance by utilizing previous range images. The proposed algorithm now supports application on a mobile robot with online measurement density enhancement for 3D range images collected by rotating a 2D lidar device around its pitch axis with a high 3D scan frequency. We show illustrative examples for different scenarios to present the effectiveness of the proposed method on a 3D range sensor mounted on a mobile robot.

Keywords

Range sensing, 3D lidar, measurement density, Papoulis–Gerchberg, iterative closest point

Introduction

Obtaining sensory information is a critical ability for mobile robots to be functional in unknown environments. Range information of surroundings is especially important for such tasks as navigation (Travis et al., 2005), obstacle avoidance (Sabatini et al., 2014) and vehicle detection and localization (Ponsa et al., 2011). Radar (Herman, 1991), stereo vision (Jung et al., 2007) and lidar (Hall, 2011) are commonly used range sensors to capture range information from the environment. Among these, lidar devices are one of the most popular range detectors in unmanned ground vehicles and mobile robotic applications, such as road boundary detection (Han et al., 2012; Wijesoma et al., 2004) obstacle detection (Darms et al., 2009; Matthies and Grandjean, 1994) and heading estimation (lidar compass) (Gallant and Marshall, 2016) in urban environments.

In recent years, with advancements in technology, commercially available 3D lidars have become an option for obtaining 3D range information of an environment. Although a variety of 3D range sensors are available, with a wide price range, the costs are still quite high for low-budget systems. Conversely, there are also applications that implement low-cost 2D lidar devices, such as geometric property estimation, feature extraction and geometric model construction (An et al., 2013). However, the use of 2D information is still an unfavourable option for autonomously moving robots, since this is reported to be insufficient for many applications,

including segmentation of different objects (MacLachlan, 2005) or detecting holes and hills or other roughness of the environment (Leonard et al., 2008). Another option is to use multiple 2D lidar devices in a system to obtain 3D range information of the environment. Thrun et al. (2006) report the use of five 2D lidar devices on their autonomously operating vehicle, which won the DARPA Grand Challenge. These five 2D lidar devices were responsible for detecting road boundaries and obstacles by scanning the environment at different

¹Department of Electronics and Communication Engineering, İstanbul Technical University, Turkey

²Department of Electrical and Electronics Engineering, Bilkent University, Turkey

³Department of Electrical and Computer Engineering, Northeastern University, USA

⁴Department of Control and Automation Engineering, İstanbul Technical University, Turkey

⁵Aselsan A.S., Akyurt Facilities, Ankara, Turkey

⁶Laboratory of Computational Sensing and Robotics, Johns Hopkins University, USA

*These authors contributed equally.

Corresponding author:

Ismail Uyanik, Laboratory of Computational Sensing and Robotics, Johns Hopkins University, Baltimore, MD 21218, USA.

Email: uyanik@jhu.edu

pitch angles. However, in this case, the total cost of the sensor system is still unaffordable and the weight may pose a problem for small-scaled systems. Therefore, the development of a low-cost and lightweight lidar system providing 3D range information of the environment is still a focus of interest.

Rotating a sensory system to increase sensing volume in an environment is a common and well-accepted method in the literature. For instance, pan tilt mechanisms are frequently used with cameras to increase their fields of view (Neumann et al., 2014; Yu and Yu, 2011). Motivated by this, an ad-hoc solution for obtaining a low-cost 3D range sensor would involve rotating a 2D lidar device around its axes. Fortunately, this solution is frequently used in the literature (Kang and Doh, 2016; Ohno et al., 2009; Surmann et al., 2001), including some other techniques, such as using a spring-mounted structure (Bosse et al., 2012) or a resonant mirror on a scanning motor (Kimoto et al., 2014) to generate 3D range information with a single 2D lidar device. Surmann et al. (2001) use a standard servomotor to rotate a 2D lidar device attached to an extended mount around a horizontal axis. Their standard resolution is 22500 points in $150(h) \times 90(v)$; it is collected within 4 seconds, which may cause an unacceptable measurement delay while moving at high speeds or detecting sudden changes in the environment (Surmann et al., 2001). Ohno et al. (2009) also generate high measurement density 3D range images by rotating a 2D lidar device around both its yaw and pitch axes with a 5–20 s 3D scan time. However, high measurement delay (due to low scan frequency) may lead to insufficient reconstruction of the environment when the robot is moving fast. Note that accurate reconstruction of a signal requires a sampling rate higher (at least two times) than the signal frequency. Adopting the same principle for our 3D range sensor on a moving robot, the 3D scan frequency (sampling rate) should be higher than the robot velocity (signal frequency) for proper representation of the environment.

Conversely, rotating a 2D lidar device to obtain 3D range data produces a resolution problem at high rotational frequencies. A common solution for increasing the resolution in image processing is to interpolate missing data. However, super-resolution of the lidar range data is still under investigation and only a few results are available in the literature. Premebida et al. (2016) interpolate 3D lidar data using a modified bilateral filter to increase the measurement density. However, the time complexity of using interpolation for increasing measurement density is considerably high ($O(m^2)$ per pixel, where m is the filter size (Weiss, 2006)) and requires the data to be uniformly distributed on a high-resolution grid to provide satisfactory results (Premebida et al., 2016). Nevertheless different implementations of bilateral filters with smaller time complexity have been developed by using some approximations, such as separable (Pham and Van Vliet, 2005) and box spatial kernels (Weiss, 2006). However, the practicality of such implementations on 3D lidar range data has not been investigated yet, to the best of our knowledge; hence, their performance in a real application is still questionable.

Motivated by these problems, our aim in this study is to present a new technique, which is capable of increasing the measurement density of 3D range images with a low time complexity. In a previous work (Özbay et al., 2015), we adopted the Papoulis–Gerchberg (P–G) algorithm (Papoulis,

Figure 1. 3D range sensor on a mobile robot with associated coordinate frames.

1975) to enhance the measurement density of single 3D range images that were collected with a 3D range sensor built by rotating a 2D lidar device. However, the slow convergence rate of the P–G algorithm limited the use of proposed method to off-line applications.

The key contribution of this paper is to develop upon the original P–G algorithm to reduce the convergence time for a continuous flow of range images by utilizing sensory information from previous frames. In addition, we present a low-cost 3D range sensor that is built by rotating a 2D lidar device around its pitch axis using a servomotor. Moreover, we mounted the 3D range sensor on a Pioneer P3-AT robot platform, which is commonly used for experimenting with lidar applications (Forbes, 2013). We also provide a method to synchronize data coming from the 2D lidar device, servomotor and mobile robot platform. We illustrate the performance of the proposed algorithm, the dynamic Papoulis–Gerchberg (dynamic P–G) algorithm, on explanatory scenarios using data collected with our 3D range sensor on a mobile robot as well as data obtained from the KITTI dataset (Geiger et al., 2013). Our results show that the dynamic P–G algorithm provides sufficiently successful results for increasing the measurement density of 3D range images in various scenarios.

3D range sensor: system description

In this section, we present the electromechanical structure of our experimental setup, along with the software architecture and 3D data collection strategy. Our 3D range sensor, as illustrated in Figure 1, operates by rotating a 2D lidar device using a servomotor to obtain 3D range data from the environment. Similar techniques for rotating a 2D lidar device to obtain 3D range data can be categorized into three groups, pitching scan, rolling scan and yawing scan, based on the axis around which the 2D lidar device is rotated (Wulf and Wagner, 2003). We use the pitching scan method in our system, owing to its advantages for full frontal coverage even with rotation angles much less than 360° .

System architecture

Rotation of the 2D lidar device is achieved through a mechanical unit consisting of a servomotor with a rotating shaft and

a bearing unit attached on opposite sides of the 2D lidar device, as illustrated in Figure 1. To ensure smooth and stable rotation, the servomotor and the bearing unit are aligned with the centre of mass of the 2D lidar device. We also built a plexiglass shell to hold all the necessary components of our 3D range sensor and form it as an independent product. This allows us to test our 3D range sensor with different robot platforms.

In terms of electronics, our 3D range sensor consists of a 2D lidar device to collect 2D range data from the environment, a servomotor to rotate the 2D lidar device, a microcontroller to drive the servomotor, a mini-computer to run the proposed algorithm and a power board to distribute power for the associated components from a 12 V battery.

Our 2D lidar device is a Hokuyo UTM-30LX device, which has a 30m maximum measurement distance and a 270° planar scan range with 0.25° resolution. The lidar collects a single scan in 0.025 s; the data are transmitted to a PC through a USB connection using the SCIP 2.0 communication protocol. The servomotor rotating the 2D lidar device is a Dynamixel RX-24F servomotor, which can sweep a rotation range of 0° – 300° with a 0.3° resolution and 126 r/min angular speed. The communication interface between the servomotor and an Arduino Mega 2560 microcontroller is developed using RS485 protocol by building MAX485 circuitry.

An Intel NUC Kit NUC5i7RYH is used as a mini-computer to collect, synchronize and merge the data coming from different sensors and to process the 3D range data. The NUC is composed of Intel core duo 3.34 GHz CPU, 16 GB RAM and a 64-bit operating system. To perform mobile tests, we placed the 3D sensor system on a Pioneer 3-AT mobile robot of 27.7 cm height and 12 kg weight with four encoders, one on-board computer and three 7.2 Ah batteries. Moreover, the mobile robot has a maximum forward or backward speed of 0.7 m/s and a rotation speed of 140°/s. Figure 2 illustrates the electronic components and their connections.

The software architecture of our experimental setup includes three different programming languages, which are specifically used for different tasks. First of all, Arduino code, which is based on the C/C++ programming language, is implemented in the microcontroller to drive the servomotor and to collect pitch data and receive *sync* signals from the 2D lidar device and the mobile robot. Secondly, Java programming language is used to gather planar range data from the 2D lidar device, pitch angle data from Arduino and position information from the mobile robot. Synchronization of sensor data to represent a point in 3D space with spherical coordinates (r, θ, ϕ) is also performed in Java. Finally, Java is used to transfer raw low-resolution data to Matlab via transmission control protocol (TCP) communication for further processing in Matlab, where the P-G and dynamic P-G algorithms are implemented.

3D data collection

In this section, our goal is to explain how a 3D range image is obtained with our 3D range sensor. To preserve consistency, we refer to a single scan of the 2D lidar device as a *layer*. Similarly, the 3D scan data, which consist of all 2D lidar layers obtained during either upwards or downwards rotation of the servomotor, are defined as a *frame* throughout the

Figure 2. System architecture and components.

paper. Note that since our 2D lidar device works at 40 Hz and our 3D scan frequency is 5 Hz, each frame includes eight layers, by definition.

A single point, P (illustrated in Figure 1), is registered to 3D space by merging information collected from different sensors. For the sake of clarity, we first explain how the point P is registered to $O'X'Y'Z'$, which is the measurement frame centred at the centre of the 2D lidar device. Then we express this point in the global coordinate frame via linear coordinate transformations. To summarize, 2D lidar represents a measured point in polar coordinates as $\mathbf{q}_p = [r \ \theta]$. Then, pitch angle, ϕ , obtained from the servomotor is used to represent the point in spherical coordinates ($O'X'Y'Z'$), as

$$\mathbf{q}_s = [r \ \theta \ \phi] \quad (1)$$

Note that the (r, θ) and ϕ information comes from different sensors, whose clocks must be synchronized in order to match the corresponding pitch angle with a 2D lidar measurement. The problem with synchronization is that there will be an inevitable offset and drift between the Arduino and 2D lidar clocks. To attack both problems simultaneously, we first connect the *sync* signal of the 2D lidar to Arduino, as shown in Figure 2. The *sync* signal generates a pulse at the beginning of each 2D scan and overcomes the increasing time offset by continuously matching the times of Arduino and 2D lidar at these instants. As an empirical solution to the drift problem, we store the ϕ values obtained between two 2D lidar pulses with their times with respect to the Arduino clock and interpolate them to generate a ϕ value for each (r, θ) . Although there are different methods of sensor synchronization (Olson, 2010), we utilized this relatively simpler solution, whose accuracy has been shown in our experiments, conducted in wide variety of scenarios, to reduce complexity in our analysis.

After obtaining a synchronized data representation for any point P in the $O'X'Y'Z'$ coordinate frame, we aim to transform this representation to the global coordinate frame, $OXYZ$, by applying the transformations in equation (2). However, we first need to match each point P with a

corresponding robot position by using a similar synchronization approach. First, we produce a *sync* signal from the internal microcontroller of the mobile robot and then transmit this signal to the Arduino. This *sync* signal is used for marking encoder data transmission from the robot to the Arduino by generating a synchronization pulse. Then, we utilize this *sync* signal, in a similar method to that explained in the previous paragraph. After completing synchronization, we transform any point P in $O'X'Y'Z'$ to the global coordinate frame, $OXYZ$, through the following transformation

$$P_g = T_r^g \ T_s^r \ T_l^s \ P \quad (2)$$

where P_g is the representation in the global coordinate frame. In Figure 1, the lidar measurement frame ($O'X'Y'Z'$) expresses the coordinate system where the lidar data is collected. Then, 3D data are transformed to the 3D sensor frame ($O_sX_sY_sZ_s$) by T_l^s , a simple transformation matrix that rotates the lidar measurement frame just to obtain a more intuitive spherical coordinate representation. T_s^r represents the transformation from the 3D sensor frame (in spherical coordinates) to the robot frame ($O_rX_rY_rZ_r$) and T_r^g represents the transformation from robot frame to global coordinates ($OXYZ$). Note that T_s^r is a fixed-valued transformation matrix that can be measured directly from the mobile robot. However, T_r^g depends on the robot position; hence, it needs to be re-evaluated for each time instant based on the new robot position. Due to space considerations and their easy-to-compute structures, the values of the transformation matrices T_l^s , T_s^r and T_r^g are not given in the current manuscript.

Note that while dealing with different sensors, besides synchronization, data loss may arise as another difficulty, owing to the fragility of connection and data transmission between sensors. In our system, we do not expect any transmission loss due to congestion, since the bandwidths of the communication interfaces connecting the sensors to the on-board computer are wide enough to carry all the data coming from the sensors at the desired transmission frequency. Moreover, the processing power of the suggested on-board computer in our system is overqualified for collecting, processing and merging data from sensors in real time. Actually, a possible future study would involve utilizing a field-programmable gate array unit, which is very efficient for fast Fourier transform or inverse fast Fourier transform operations, to reduce the equipment cost with a hard real-time electronic infrastructure. Apart from this, although wireless communications between an on-board computer and a client computer are prone to data loss, our choice of TCP as a reliable protocol – it prevents data loss by resending it in cases of transmission failures – allows our 3D sensor system to be robust against data loss. In addition to these, to avoid any delay during processing in Matlab, which can cause transferred data on the client computer to be lost, we tried to shorten transmission and processing times by minimizing the quantity of the transferred data. For this purpose we encoded the data according to the lidar communication protocol, SCIP 2.0 (scanning sensor command interface protocol) (Kawata et al., 2005). We have also used the same encoding protocol while communicating with the servomotor and transferring data through wireless communication.

Validation of 3D registration

In this section, our goal is to evaluate the 3D registration performance of our data collection strategy. To accomplish this, we adopt a solution from the literature (Noguchi et al., 2012), where 3D registration performance is evaluated on flat wall data by quantifying deviations from the wall as an error metric.

In this experiment, we aim to validate our 3D range sensor's point registration capability using the P–G algorithm on a flat wall, which is mainly used to observe the consistency of the registered points on a planar surface as a ground truth. Hence, the 3D range sensor is placed stationary 5 m away from a flat wall and 100 frames of data for $11^\circ(h) \times 13^\circ(v)$ area are collected within 20 s to reconstruct the wall in 3D space. Then, a flat surface is fitted to the measured data, resulting in a distance of $y^j = 5.007$ m, which is consistent with our manual measurements (5 m). To evaluate the performance of our 3D registration method, we define two error metrics, the mean error and the maximum error, as

$$E_{\text{mean}} = \frac{1}{M} \sum_{m=1}^M |y_m - y^j| \quad (3)$$

$$E_{\text{max}} = \max(|y_m - y^j|, \forall m \in \{1, 2, \dots, M\}) \quad (4)$$

where M is the number of measured data points and y_m corresponds to the distance measured for the m th point. The resulting errors for our flat wall experiment are obtained as

$$E_{\text{mean}} = 5.035 \text{ mm} \quad E_{\text{max}} = 16.184 \text{ mm}$$

Note that the resulting errors stay well inside the accuracy of our 2D lidar system, which is reported as ± 30 mm. Therefore, as can be seen from the demonstrated results, our data collection strategy does not introduce more error than commercial 2D lidar systems already have. Furthermore, this accuracy is also in a range that is comparable to the reported accuracies of commercially available lidar devices used for 3D imaging, such as Reigl VUX-IHA, Reigl VUX1, Scansweep v1.0, Velodyne 3D lidars HDL32E, HDL64E and Velodyne Puck, with respective accuracy errors of ± 5 mm (RIEGL Laser Measurement Systems GmbH, 2017), ± 10 mm (RIEGL Laser Measurement Systems GmbH, 2017; Scansweep LLC, 2014), ± 20 mm (Velodyne LiDAR, Inc., 2012, 2016a) and ± 30 mm (Velodyne LiDAR, Inc., 2016b).

Increasing measurement density via dynamic Papoulis–Gerchberg algorithm

The main aim of this study is to enhance the measurement density of an originally low measurement density range image collected by a 3D range sensor with high 3D scan frequency. There is a natural trade-off between measurement density and 3D scan frequency for 3D range sensors that are based on rotating a single 2D lidar device. Since we seek for high 3D scan frequency (5 Hz), we obtain low measurement density range images, i.e. eight layers per 3D frame. Thus, we aim to enhance measurement density via online post-processing of 3D range data. To accomplish this, we utilize a super-

resolution algorithm, Papoulis–Gerchberg (P–G), to increase the measurement density of our 3D range images.

Although the literature reports studies using the P–G method in various fields, such as software-defined radio systems (Magalhaes et al., 2013), it is mainly used in image processing for super-resolution and recovering lost samples of an image (Salari and Bao, 2012) based on iterative implementation of fast Fourier transform and inverse fast Fourier transform operations. Chatterjee et al. (2009) also present the application of the P–G algorithm to perform image super-resolution and inpainting of low-resolution images. Similarly, discrete finite-dimensional versions of the well-known P–G extrapolation algorithm are implemented in Ferreira (1994) to recover missing samples in the finite-length records of band-limited data. In this respect, our goal for using this algorithm is to enhance the measurement density of a 3D range image while working at a high 3D scan frequency by estimating unknown points using measured range information from the environment.

The proposed method starts with the projection of a 3D range image, represented as a point cloud in 3D spherical coordinates (r, θ, ϕ) , to a high-resolution 2D grid according to their θ and ϕ values, where r represents the value at the corresponding (θ, ϕ) index. The resolutions of the 2D grid in the θ and ϕ directions are determined based on the resolution of the 2D lidar device (0.25°) and the servomotor (0.3°), respectively. As an example, for a 3D range image with 90° horizontal and 30° vertical range, we obtain a high-resolution 2D grid $\in \mathbb{R}^{m \times n}$, where $m = 100$ and $n = 360$. Unlike Premebida et al. (2016), the P–G algorithm does not require a condition on data distribution and we have the opportunity to apply the P–G algorithm to specific parts of an image. For instance, in this paper, we apply the dynamic P–G algorithm in the vertical direction only, since 2D lidar outputs sufficiently dense data in the horizontal direction.

Dynamic P–G algorithm: application to 3D range data

Although the iterative P–G algorithm is well known to recover the lost samples of a signal, the convergence rate of the algorithm can be insufficient for online applications (Papoulis, 1975). A main factor affecting the convergence time of the P–G algorithm is the number of lost samples in a signal. Considering our sparse range data matrices, we seek to reduce the convergence time using estimates from previous measurements.

Like the approach proposed in this paper, there are several studies in the literature that extend the standard P–G algorithm to speed up the convergence rate (Chamzas and Xu, 1984; Sun et al., 2015; Xu and Chamzas, 1983). For instance, Chamzas and Xu (1984) propose a novel method to increase the convergence rate of the P–G algorithm. They introduce a recursive multiplication term, which is chosen to minimize the error between the actual and estimated signals by multiplying the estimates at each iteration. Moreover, Sun et al. (2015) propose a modified P–G algorithm, similar to that of Chamzas and Xu (1984), by introducing another iterative process to gradually reduce the influence of noise via additional multipliers. The utility of this algorithm has been shown for tomography image reconstruction from limited

angle projections (Sun et al., 2015). However, such methods are all based on the utilization of a single image of the environment and introducing some optimization processes to speed up the convergence process. Hence, these approaches are not preferable for our purposes, since we aim to exploit the P–G algorithm to handle a flow of images and we try not to introduce any complexity to the standard P–G algorithm.

In image processing, the P–G algorithm is mainly used for super-resolution of a single image. However, in our case, we have a continuous flow of range images, where sequential frames contain many common points, as an advantage of our high 3D scan frequency. Originally inspired by its image processing applications, we now extend the P–G algorithm by utilizing estimates from the previous frame (only those points that are common with the current frame) as an initial condition for the unmeasured points. This new process is called the dynamic P–G algorithm.

Before starting a detailed description of the dynamic P–G algorithm, we first give some definitions that will ease the understandability of the proposed method. Note that the variables defined are all two-dimensional matrices whose frame numbers are represented with the index k .

- $X[k] \in \mathbb{R}^{m \times n}$. A virtual high-resolution matrix containing real environment range data.
- $M[k] \in \mathbb{R}^{m \times n}$. A measurement matrix with binary coefficients. Any index with value 1 corresponds to a measured point from the environment, while indices with value 0 represent unmeasured points.
- $V[k] \in \mathbb{R}^{m \times n}$. The measured portion of $X[k]$, such that

$$V[k] = M[k] \circ X[k] \quad (5)$$

where $\circ$ represents the Hadamard product operator, which provides element-wise multiplication for two matrices (see Remark 1).

- $\bar{V}[k] \in \mathbb{R}^{m \times n}$. The estimated environment matrix with P–G estimates for unmeasured points.
- $g(\cdot)$. A function that returns a binary matrix indicating non-zero elements with 1, leaving the others as 0.

Remark 1. Note that $X[k]$ is a virtual matrix, since full resolution range information of the real environment is actually not available. However, we define $X[k]$, $M[k]$ and equation (5) to describe projecting a measured frame to a high-resolution grid in a mathematical sense for further use in our algorithm.

To start with, new data, $V[k]$, obtained from the 3D range sensor for the k th frame, are projected on a high-resolution grid, as in equation (5). Note that the dynamic P–G operation described next is repeated for each measured frame, $V[k]$, for $k = 1, \dots, N_f$, where N_f is the number of frames. The remaining part of this section explains the details of the proposed algorithm, which runs on an iterative basis to estimate $\bar{V}_i[k]$ for each step, i , until an optimal estimation, $\bar{V}_{\text{opt}}[k]$, is achieved.

Step 0. As an initialization procedure, we transfer the P–G results of the previous frame, $\bar{V}_{\text{opt}}[k-1]$, to the current frame, $V[k]$, via a coordinate transformation to find the

corresponding indexes in the current frame. We initialize the unmeasured points in these indexes with estimates from the previous frame. For the remaining unmeasured points, we assign a maximum depth, since 2D lidar returns the maximum range in the case of no detection. The initialization process is performed as

$$\begin{aligned}\bar{V}_0[k] &= V[k] \\ &+ (\underline{1} - M[k]) \circ (W[k]) \\ &+ [(\underline{1} - M[k]) \circ (\underline{1} - g(W[k]))] \cdot d_{\max}\end{aligned}\quad (6)$$

where $W[k] = T[k]\bar{V}_{\text{opt}}[k-1]$ and $T[k]$ is a transformation matrix for transferring points from the previous frame, $\bar{V}_{\text{opt}}[k-1]$, to their corresponding indices in the current frame $V[k]$. Note that $T[k] = 0$ and $\bar{V}_{\text{opt}}[k-1] = 0$ for $k = 1$ (see Remark 2).

Remark 2. Note that the initialization procedure, Step 0, is not performed for the original P-G algorithm, where the missing points are assigned as 0.

Step 1. $\bar{V}_i[k]$ is filtered using a low-pass filter to eliminate high-frequency elements in the measurement:

- Perform one-dimensional (in the direction of ϕ) discrete Fourier transform:

$$\bar{V}_i(\omega) = \mathcal{F}\{\bar{V}_i[k]\} \quad (7)$$

- Apply low-pass filter, $\mathcal{L}_{w_c}$, with a normalized cut-off frequency of 0.1, which is determined based on the data histogram:

$$\bar{P}_i(\omega) = \mathcal{L}_{w_c}\{\bar{V}_i(\omega)\} \quad (8)$$

- Perform one-dimensional (in the direction of ϕ) inverse discrete Fourier transform:

$$\hat{V}_i[k] = \mathcal{F}^{-1}\{\bar{P}_i(\omega)\} \quad (9)$$

Step 2. Compute the percentage iteration error ξ for the i th iteration as

$$\xi_i = \frac{\|M[k] \circ X[k] - M[k] \circ \hat{V}_i[k]\|_2}{\|M[k] \circ X[k]\|_2} \times 100 \quad (10)$$

Step 3. Force the original measurements into the estimated matrix

$$\bar{V}_{i+1}[k] = M[k] \circ X[k] + (\underline{1} - M[k]) \circ \hat{V}_i[k] \quad (11)$$

where $\underline{1}$ is a matrix of ones.

The iterations from Step 1 to Step 3 will continue until

$$|\dot{\xi}| < \xi_{\min} \quad (12)$$

where $\xi_{\min} = 10^{-3}$, corresponding to a threshold value to determine error convergence, which is chosen based on

previous observations. Note that the convergence of equation (10) has been proven in Papoulis (1975).

To prevent any erroneous estimations, such as ripples and outliers, a first-derivative filter is applied in the vertical direction to eliminate points whose derivatives are greater than 30 mm (selected based on 2D lidar accuracy). After setting eliminated points to infinity (NaN in Matlab), the resulting matrix is determined as $\bar{V}_{\text{opt}}[k]$ for the k th frame.

Discussion of convergence rate of dynamic P-G algorithm

This section is devoted to a discussion on how the dynamic P-G algorithm reduces the convergence time of the standard P-G algorithm. In the previous section, we explained the mathematical formulation of the dynamic P-G algorithm, which is applicable to 3D lidar range data. However, we need to introduce the key differences between the dynamic P-G and P-G algorithms to emphasize the improved convergence rate in the dynamic P-G algorithm.

The key logic behind the P-G algorithm comes from the smoothness assumption of the physical world. Basically, when a signal with lost samples is given as an input to the P-G algorithm, it assigns zero for missing samples and tries to smooth the signal through an iterative process. To accomplish this, the P-G algorithm first transforms the system to the frequency domain via Fourier transform such that the impulsive affect of zero values (assigned for missing samples) appear as sinusoidal artefacts that are spread over the whole frequency band. The P-G algorithm filters these artefacts with a low-pass filter and tries to reconstruct the original smooth signal with accurate estimates of the missing samples. Note that there will always be some artefacts in the low-frequency regime, since the low-pass filter cannot eliminate them. More importantly, such artefacts in the low-frequency regime will also bring some error for the prediction of the measured points. However, the P-G algorithm imposes the original measurements on the results after each iteration to ensure that estimation can only update the missing samples, keeping the measurements the same through the whole process.

A long-debated issue related to the convergence rate of the P-G algorithm is the effect of the assignment of zero values for missing samples (Chamzas and Xu, 1984; Hsu and Lo, 2005) in the signal. Since these impulsive zero values contaminate the whole frequency band, yielding a huge prediction error, they are a main reason for the slow convergence rate of the standard P-G algorithm. Several works in the literature focus on the introduction of additional interpolation methods to find better initial guesses for the missing samples (Henrion et al., 2004), since any value closer to the overall signal trend would do a better job in terms of reducing the initial prediction error and hence the convergence time. However, such methods increase the complexity of the algorithm with additional interpolation steps, which prevent high-frequency real-time applications.

One important point here is that the standard P-G algorithm (Papoulis, 1975), as well as many of its variants (Chatterjee et al., 2009; Ferreira, 1994; Salari and Bao, 2012) are designed to work with a single measurement of a signal over a finite horizon, which is an image in most cases.

However, the applications we are interested in, such as navigation of mobile robots with sensors, allow continuous flow of high-frequency measurements that might have a considerable intersection area. The key idea of the dynamic P-G algorithm is to utilize this continuous flow of measurements to find an optimal initial guess for the missing samples in a signal, since we know that a better initial guess has a dramatic effect on the convergence rate.

The way that dynamic P-G algorithm uses the continuous flow of range images is that it first transforms the previous range image (including estimates of the missing samples) onto the current range image to find any possible intersections. Then, the dynamic P-G algorithm initializes the missing samples of the current image with the values from the previous range image. There are two key points that deserve a deeper discussion here. First, the high 3D scan frequency of the proposed 3D lidar system ensures that there is a great intersection between two consecutive range images, even if the mobile robot moves fast. Hence, the dynamic P-G algorithm starts with a much denser signal than the standard P-G algorithm. Second, the dynamic P-G algorithm uses the values from the previous image only as an initial guess of the missing samples without forcing them as actual measurements. This is the key for ensuring the applicability of the proposed algorithm in a dynamic environment, such as the moving-person example given in the following sections.

Note that, apart from the key difference in the initialization process, the mathematical formulation and the operations of the dynamic P-G algorithm are almost the same as those of the standard P-G algorithm. Hence, it is fair to say that they have a similar time complexity in a mathematical sense (the complexity of applying sequential fast Fourier transform and inverse fast Fourier transform operations). However, the smart choice of initial conditions for the lost samples plays a fundamental role in convergence time, yielding a great advantage for the dynamic P-G algorithm.

One should note that the convergence and accuracy for the standard P-G algorithm has already been proved (Papoulis, 1975). Hence, utilizing P-G estimates from the previous step can be considered as a key choice for the initialization of lost samples. Such accurate initial conditions, which preserve the general smoothness of the data (estimated to guarantee smoothness via the P-G algorithm in the previous measurement), yield small impulsive noises and hence smaller contamination in the frequency spectra. Hence, filtering and reconstructing the signal yields a much smaller prediction error with such initial conditions starting from the first iteration. This principle is the idea behind the improved convergence rate of the dynamic P-G algorithm with respect to the standard P-G algorithm. Mathematically speaking, assume that we have a signal $\bar{x}[n]$ given as

$$\bar{x}[n] = \begin{cases} A_1, & \text{for } n = 0, 1, 2, 3, 4, 5, 6, 8, 9 \\ 0, & \text{for } n = 7 \end{cases} \quad (13)$$

The coefficients of the frequency components after Fourier transform can be computed using

$$\bar{a}[k] = \frac{1}{10} \sum_{n=0}^9 \bar{x}[n] e^{-j2\pi k n / 10} \quad (14)$$

which can be solved numerically for the signal given in equation (13) as

$$|\bar{a}[k]| = \begin{cases} \frac{9A_1}{10}, & \text{for } k = 0 \\ \frac{A_1}{10}, & \text{for } k = 1, 2, 3, 4, 5, 6, 7, 8, 9 \end{cases} \quad (15)$$

Note that the effect of the impulsive zero value assigned for the lost sample in equation (13) contaminates the whole frequency band, as seen in equation (15). Application of the standard P-G algorithm on this signal yields the following estimate for $\bar{x}[n]$ as

$$\hat{x}[n] = \begin{cases} A_1, & \text{for } n = 0, 1, 2, 3, 4, 5, 6, 8, 9 \\ A_2, & \text{for } n = 7 \end{cases} \quad (16)$$

where $\|A_1 - A_2\| < \epsilon$ by the convergence of the standard P-G algorithm.

Since we have a continuous flow of these signals, assume that we measure the same signal in equation (13) again for simplicity of discussion. The key difference is that we now have a better estimate of the missing sample at $\bar{x}[7]$ from equation (16). Therefore, we can initialize $\bar{x}[7] = A_2$ before starting the P-G process. Now, magnitudes of the Fourier series coefficients of this signal will take the form

$$|\hat{a}[k]| = \begin{cases} \frac{(9A_1) + A_2}{10}, & \text{for } k = 0 \\ \frac{A_1 - A_2}{10}, & \text{for } k = 1, 2, 3, 4, 5, 6, 7, 8, 9 \end{cases} \quad (17)$$

Note that the prediction error, as defined in equation (10), considers the difference between the actual measurements and their predictions. To make our point clear for this constant signal example, we can relate the prediction error to the error in the DC component, since we would expect all power to be accumulated on DC for a constant signal in the form of equation (13). Therefore, while the initial error of the standard P-G algorithm is $A_1/10$, it reduces to $\epsilon/10$ with the dynamic P-G algorithm. This initialization process is the key idea behind the improved convergence rate of the dynamic P-G algorithm.

Investigating dynamic P-G estimates: 3D registration and convergence performance

In previous sections, we showed that the points registered to a 3D frame by our 3D range sensor are consistent for a flat wall data. Now we investigate the consistency of estimated points by the dynamic P-G algorithm using the same error metrics reported in equations (3) and (4). However, this time the robot starts from a 10 m distance and moves towards the flat wall. A sample frame is chosen when the robot is 5 m away from the wall for a fair comparison with our previous tests. The results for the dynamic P-G algorithm are

$$E_{\text{ave}} = 9.729 \text{ mm}, \quad E_{\text{max}} = 19.742 \text{ mm} \quad (18)$$

which are still within 2D lidar resolution accuracy (see Remark 3).

Remark 3. Notice that filtering ripples and outliers (points whose derivative is more than 30 mm) eliminates 1.8% of

Figure 3. Comparison of convergence rates of Papoulis–Gerchberg (P–G) and dynamic P–G algorithms for sample data.

total estimated points for this test. Also note that this filtering process does not conflict with our error metrics defined in equations (3) and (4), since the filtering operator eliminates points that are wide apart from smooth surfaces. Therefore, the maximum error, $E_{\max} = 19.742$ mm, computed in equation (18) states that there is no point cluster beyond this value.

Having validated the consistency of the points estimated using the dynamic P–G algorithm, we now investigate the practicality of our claim to reduce convergence time by the use of previous frame estimates as the initial condition. Figure 3 illustrates the convergence behaviour of the same frame (obtained when the robot is 5 m away from the wall) for both P–G and dynamic P–G algorithms. As expected, using previous estimates as the initial condition in the dynamic P–G algorithm drastically reduces convergence time by initializing the process with a much smaller prediction error in the first iteration, as explained in the previous section. Based on equation (12), the convergence time is reduced to 50 iterations in the dynamic P–G algorithm, from 300 for the P–G algorithm. This result also reduces the computation time of the P–G algorithm, which now allows online implementation with a 3D scan frequency of 5 Hz. For example, dynamic P–G operation takes 30 ms on a standard $90^\circ(h) \times 30^\circ(v)$ data, while original P–G operation takes 207 ms.

Results and discussion

This section details some illustrative examples from various scenarios in which we increase the measurement density of 3D range images using the dynamic P–G algorithm. For clarity, actual measurements by the 3D range sensor are represented in red while dynamic P–G estimates are indicated in blue throughout the section.

Detection of moving objects with high 3D scan frequency

This section aims to illustrate the importance of high 3D scan frequency, as well as the need for high measurement density, to detect moving objects during their motion in an environment. To illustrate our point, we performed two experiments for detecting person walking through our 3D range sensor's view at an (almost) constant velocity with the stationary

robot. Although both tests were performed for the same duration (1 s), the first one was performed with a scan frequency of 5 Hz (fast), while the second one was performed with a scan frequency of 1 Hz (slow). Note that the slow method results in only one high measurement density frame for the whole motion of the person (Figure 4(b)), while the fast one receives five low measurement density frames (Figure 5).

The important point that needs to be discussed about Figure 4(b) is that the slow method, with a scan frequency of 1 Hz, is not capable of capturing spatiotemporal characteristics of human movement, owing to its high time constant. Instead, it detects different parts of the person's body over time, yielding the tilted person shape given in Figure 4(b). We see the head of the person first, possibly since the 3D range sensor started scanning from its highest pitch position. Then, the sensor slowly scans downwards over the whole body. However, since the person walks in the meantime, the other parts of the body are detected as if the person is tilted, which is not the case in our experiments. This is a sample fundamental example, demonstrating the need for a high 3D scan frequency for mobile robotic applications, especially in the existence of dynamic objects in the environment.

Conversely, the fast scan method, with a scan frequency of 5 Hz, yields a much better illustration of the spatiotemporal characteristics of the walking person, as seen in Figure 5. For the same time interval (1 s), the fast method captures five different range images of the walking person, yielding a better spatiotemporal observation of the environment. However, note that the fast method yields low measurement density range images with respect to the slow one, yielding only the red dots illustrated in Figure 5. Hence, to improve the measurement density of these images, we apply the dynamic P–G algorithm and estimate the blue points illustrated in Figure 5 as a better representation of the moving person with an increased measurement density. For a better comparison, a sample scene from the fast scan case (which is collected in 200 ms) is depicted in Figure 4(a). Note that both images in Figure 4 have similar measurement density but the one on the left represents a better spatiotemporal detection performance, since its scan frequency was much higher.

Experiments with a moving robot in a static environment

In this section, we describe experiments with a moving robot in a static environment to illustrate the practicality of dynamic P–G algorithm for online applications on a mobile robot. Since there will be a continuous flow of range images during the motion of the robot, we will illustrate the results through a 3D registration process in global coordinates as a first step towards simultaneous localization and mapping (SLAM) applications. Note that most SLAM applications do not need a static environment assumption but we leave this case out of the scope of this paper, since it deserves and requires more detailed work on SLAM methods. However, simultaneous localization based on range measurements is still needed, since encoders are not successful enough for accurate localization of the mobile robot.

Figures 4. Result of dynamic P–G algorithm for: (a) single-frame walking person data collected at 5 Hz; (b) single-frame walking person data collected at 1 Hz.

Figures 5. Results of dynamic P–G algorithm applied on five consecutive frames of a walking person; data collected at 5 Hz.

The iterative closest point (ICP) algorithm is a popular method used for accurate localization of mobile robots for registering sequential range images to a global coordinate frame. Regarding this purpose, we developed a 3D registration methodology for a mobile robot by using a 3D ICP algorithm to localize and register range images into correct positions in global coordinates. The general flow of the 3D registration methodology is shown in Figure 6. Here, $P[k-1]$

denotes the corrected position estimate of the mobile robot for the previous frame ($\bar{V}_{\text{opt}}[k-1]$) based on the ICP algorithm. Then, another low measurement density range image, $V[k]$, is obtained at the current robot position, $\hat{P}[k]$, computed as

$$\hat{P}[k] = P[k-1] + \Delta P^{\text{enc}} \quad (19)$$

where ΔP^{enc} represents position change with respect to the encoder. As stated in equation (6), dynamic P–G operation uses estimates from the previous frame, $\bar{V}_{\text{opt}}[k-1]$, by transferring the common points to the current frame via a coordinate transformation, T , which is performed based on ΔP^{enc} . Then, a high measurement density range image, $\bar{V}_{\text{opt}}[k]$, is obtained by applying dynamic P–G operation.

The final part in our 3D registration process is to apply the ICP algorithm to estimate the actual position of the robot, $P[k]$, for the current frame. To accomplish this, we feed the ICP algorithm with high measurement density range images of the previous and current frames to estimate the corrected change in the robot position between the two frames as

$$\Delta P^{\text{icp}} = \text{ICP}(\bar{V}_{\text{opt}}[k], \bar{V}_{\text{opt}}[k-1], \Delta P^{\text{enc}}) \quad (20)$$

where ΔP^{enc} is used as an initial condition in the ICP operation. Then, the corrected robot position of the current frame, $\bar{V}_{\text{opt}}[k]$, is computed as

$$P[k] = P[k-1] + \Delta P^{\text{icp}} \quad (21)$$

Figure 6. Process of 3D registration into global coordinates with consecutive use of dynamic Papoulis–Gerchberg (P–G) and iterative closest point (ICP) algorithms.

Investigating ICP robustness with dynamic P-G. Notice that our goal was to support 3D range detection for high-velocity applications, which require high 3D scan frequency. However, the dynamic P-G algorithm requires accurate estimation of the mobile robot's position, since it combines current frame data with the previous frame based on their positions. To achieve this, we utilize the ICP algorithm to estimate the mobile robot's position based on encoder data as well as the 3D range data obtained through the dynamic P-G algorithm.

This section investigates the use of dynamic P-G outputs for ICP operations and presents a comparison with ICP results obtained using low measurement density range data. To accomplish this, we designed an experiment in which the mobile robot moves in a corridor at its maximum speed (0.7 m/s) in order to register 3D range data of the environment in a global coordinate frame. Note that this application is important both to ensure that the dynamic P-G algorithm is capable of accurately estimating the robot's position (a fundamental component for projection operation in dynamic P-G) and to test its applicability for mapping applications. In this experiment, our goal is to allow the robot to move in an unknown environment with walls and obstacles and create a 3D registration of the total scene it experiences. The leftmost image in Figure 7 illustrates the 3D registration performance in this experiment (using the 3D ICP algorithm) both with the original raw low measurement density range images (red dots) and with the enhanced measurement density range images obtained using the dynamic P-G process (blue dots). Note that both sets of data produce very similar 3D registration results for the maximum speed of our mobile robot. To simulate faster movements of the robot for the same data, we select only one frame for each patch of two consecutive frames ($N_s = 2$) and apply the same 3D ICP registration process to obtain the results illustrated in Figure 7 (marked $N_s = 2$). We repeated this procedure for $N_s = 3$, $N_s = 4$ and $N_s = 5$ to mimic the robot's movement with increased speeds

and simulate possible faster tests with an unmanned ground vehicle. The key result here is that the 3D ICP process requires dense data for increased speeds, since the correlation between the two low measurement density images decreases with increased position change. Hence, when we use raw low measurement density range images for 3D registration in a global coordinate frame, we notice the mapping problems illustrated in Figure 7 using the red dots. However, as clearly shown in Figure 7 by the blue dots, the ICP-based 3D registration of the environment using the dynamic P-G results remain consistent for increased robot velocities. This is a key example, illustrating the need for increased measurement density for the 3D range images towards mapping applications.

Performance evaluation on Velodyne data. In this part, we present an experimental result for the dynamic P-G algorithm on sample data obtained from the KITTI dataset (Geiger et al., 2013) to show the performance of our algorithm independently of our 3D range sensor and mobile robot. The 3D range data in the KITTI dataset are obtained via Velodyne 3D lidar, which provides a 360° range view of the environment. In our sample data, a vehicle equipped with a Velodyne and GPS moves in an urban environment to collect a 360° range image of the environment. To make these data consistent with our experiments, we limit the horizontal measurement range to 90° and use GPS information instead of robot encoder data. Figure 8(a) depicts a single scene in the sample dataset. Although the original database includes corresponding optical images of the environment for each range data, we will only be using the 3D range images from the Velodyne sensor. Figure 8(b) illustrates the corresponding 3D range image of the scene depicted in Figure 8(a) with a Velodyne 3D lidar. Note that the scanning outputs of these types of expensive Velodyne lidar devices are already in high resolution, providing a high measurement range image of the environment. However, our goal is to mimic the case where we use a low-

Figure 7. Comparing robustness of iterative closest point algorithm for dynamic Papoulis–Gerchberg results (blue) and the original low measurement density data (red) against increasing robot velocity.

Figure 8. Application of dynamic Papoulis–Gerchberg (P–G) algorithm on sample data from KITTI dataset: (a) image of sample scene; (b) high measurement density 3D range data obtained with Velodyne; (c) downsampled Velodyne data; (d) dynamic P–G results applied on low measurement density Velodyne data.

Figure 9. 3D registration results (with iterative closest point algorithm) of dynamic Papoulis–Gerchberg algorithm on the sample data from the KITTI dataset.

cost 3D range sensor, similar to the one presented in this paper, with a low measurement density. Hence, we downsample the original Velodyne range data to imitate data collected from a low measurement density 3D range sensor. Thus, we take only eight layers per frame (as in our 3D range sensor) and place them on a high-resolution grid, as illustrated in Figure 8(c). Having obtained low measurement density 3D range images, we apply the dynamic P–G algorithm to estimate the missing samples of the original Velodyne range image and present the results in Figure 8(d). As expected, the dynamic P–G algorithm reasonably increases the density of the low-resolution (downsampled) Velodyne range images in a complex environment and provide similar high-resolution range data, as illustrated in Figure 8(d) to the original high-resolution Velodyne data shown in Figure 8(b).

Having clarified the way we are going to utilize the Velodyne data from the KITTI dataset, as illustrated in Figure 8, we now work in a continuous flow of Velodyne images while the unmanned ground vehicle moves along the yellow trajectory depicted in Figure 9. The goal here is to utilize the 3D ICP registration process (as illustrated in Figure

6) during the movements of the unmanned ground vehicle, such that we can generate 3D point clouds of the environment in a global coordinate frame using the downsampled 3D range data of the Velodyne sensor. Instead of the encoders in the mobile robot application, we now utilize the GPS data provided in the KITTI dataset, both as an initial condition in a global coordinate frame and as location information of the unmanned ground vehicle while it moves in the environment. Note that to mimic our 3D sensor, we apply the dynamic P–G operation to only a 90° part in the frontal view of the whole 360° range image. A sample illustration of this 90° view is given in Figure 9, marked as a grey shaded region. Note that this view changes based on the heading of the unmanned ground vehicle during its movement in the environment. For illustration purposes, we utilize the whole 360° for 3D registration of the environment, although the dynamic P–G algorithm works only for the 90° frontal view. The resulting 3D registration of the environment in a global coordinate frame is illustrated in Figure 9. The colour gradient from brown to green indicates increasing height, which provides a nice illustration for the trees in the environment. One can also compare the sample range view in the grey shaded region of Figure 9 with the optical image of Figure 8(a) for a better understanding of the environment and the resulting 3D registration.

For comparison purposes, we also repeated the 3D registration process for the original high measurement density Velodyne 3D lidar data. Our goal here is to compare the resulting 3D registration performance of Velodyne with the results we obtained with the low measurement density 3D range images that were supposedly collected from a low-cost 3D range sensor. For clarity of discussion, we show the 3D registration results for the red circled region in Figure 9, which corresponds to a corner of the wall, where it is easier to understand any artefacts on straight lines. Figure 10 illustrates 3D registration results of the same region obtained both by using the dynamic P–G algorithm with the ICP

Figure 10. Comparing 3D registration results of the dynamic Papoulis–Gerchberg (P–G) with the iterative closest point (ICP) algorithm (blue), Velodyne with ICP (green) and Velodyne with GPS (red).

algorithm (blue) and using the original high measurement density Velodyne data with the ICP algorithm (green). In addition, we also provide 3D registration of the original Velodyne data by using GPS locations without using the ICP algorithm (red) to emphasize the need for the ICP algorithm for better 3D registration. Figure 10 shows that Velodyne with GPS yields thicker boundaries, meaning that the 3D registration process is not successful as compared with the other two methods. However, dynamic P–G with the ICP algorithm and Velodyne with ICP algorithm produce similar 3D registration results. Note that we achieve similar performance with high measurement density Velodyne data by applying the dynamic P–G algorithm to a low measurement density range image. This is an illustration of how we can obtain similar 3D registration performances with low-cost low measurement density range sensors in a complex environment by utilizing the dynamic P–G algorithm.

A final discussion that we need to make here is about the variability of the experiments we demonstrated in this work. In general, although the proposed algorithm theoretically and intuitively proposes huge advancements to the standard P–G algorithm, this application deserves and requires detailed experimental analysis in different scenarios to demonstrate the generality of the results. To accomplish this, we tested our algorithm in many different scenarios, such as validation on a static wall (to have a ground truth environment), detection of a moving person and 3D point registration with the ICP algorithm to illustrate the performance in a dynamic environment. More importantly, to avoid any bias related to our experimental setup, we demonstrated the use of our algorithm on data taken from the well-known KITTI dataset. Hence, we believe that the dynamic P–G algorithm will be a key tool in range image processing applications, owing to its convenient use in different platforms and with different sensors.

Conclusions

This paper is a follow-up of our recent study (Özbay et al., 2015), in which we showed how an image super-resolution algorithm, P–G, can be utilized to increase the measurement density of range images. In this paper, we extend the P–G algorithm to a dynamic version by using estimates from previous frames to increase performance and reduce convergence time. We also developed a 3D range sensor with full sensor synchronization to make it applicable for mobile robot

applications. Validation experiments on a flat wall showed that both our 3D range sensor and the dynamic P–G algorithm produce consistent results with 2D lidar’s accuracy.

As a first test, we illustrated the importance of high 3D scan frequency for accurate detection of a moving person in an environment. Comparison with a low 3D scan frequency case showed that application of the dynamic P–G algorithm produces a more accurate representation of a moving person with a similar measurement density. We then applied the dynamic P–G algorithm in a moving robot experiment to investigate the robustness of the ICP method with respect to increasing robot velocities. Our results showed that increasing measurement density using the dynamic P–G algorithm produces more robust frames for 3D registration with the ICP algorithm as compared with low measurement density range images. Finally, we applied the dynamic P–G algorithm to sample data obtained from the KITTI dataset to illustrate the generality of our results. To accomplish this, we first downsampled the original Velodyne range images and then applied our algorithm to re-increase the measurement density. We then applied the 3D registration process with the ICP method to obtain a 3D representation of the environment. Our tests showed that applying the ICP method on the dynamic P–G algorithm produces 3D registration results similar to those obtained by applying the ICP method on the original Velodyne data.

Acknowledgements

We thank Professor Dr. Orhan Arikan and Professor Dr. Ömer Morgül for their invaluable ideas and support.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the Scientific and Technological Council of Turkey (TÜBİTAK) and Aselsan A.Ş.

References

- An Y, Zhuang Y, Hu H, et al. (2013) Two-dimensional laser-based environment exploration and recognition for service robots. *Transactions of the Institute of Measurement and Control* 35(8): 1068–1084.
- Bosse M, Zlot R and Flick P (2012) Zebedee: design of a spring-mounted 3-D range sensor with application to mobile mapping. *IEEE Transactions on Robotics* 28(5): 1104–1119.
- Chamzas C and Xu W (1984) An improved version of Papoulis–Gerchberg algorithm on band-limited extrapolation. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32(2): 437–440.
- Chatterjee P, Mukherjee S, Chaudhuri S, et al. (2009) Application of Papoulis–Gerchberg method in image super-resolution and inpainting. *The Computer Journal* 52(1): 80–89.

- Darms MS, Rybski PE, Baker C, et al. (2009) Obstacle detection and tracking for the urban challenge. *IEEE Transactions on Intelligent Transportation Systems* 10(3): 475–485.
- Ferreira PJS (1994) Interpolation and the discrete Papoulis–Gerchberg algorithm. *IEEE Transactions on Signal Processing* 42(10): 2596–2606.
- Forbes JR (2013) Adaptive approaches to nonlinear state estimation for mobile robot localization: an experimental comparison. *Transactions of the Institute of Measurement and Control* 35(8): 971–985.
- Gallant MJ and Marshall JA (2016) Two-dimensional axis mapping using lidar. *IEEE Transactions on Robotics* 32(1): 150–160.
- Geiger A, Lenz P, Stiller C, et al. (2013) Vision meets robotics: the KITTI dataset. *The International Journal of Robotics Research* 32(11): 1231–1237.
- Hall D (2011) *High definition lidar system*. Patent US7969558B2, USA.
- Han J, Kim D, Lee M, et al. (2012) Enhanced road boundary and obstacle detection using a downward-looking lidar sensor. *IEEE Transactions on Vehicular Technology* 61(3): 971–985.
- Henrion S, Mailhes C and Castanié F (2004) Transmitting critical biomedical signals over unreliable connectionless channels with good QoS using advanced signal processing. *WSEAS Transactions on Communications* 3(2): 694–699.
- Herman M (1991) *Electronically scanning vehicle radar sensor*. Patent US5008678A, USA.
- Hsu CY and Lo TM (2005) Improved Papoulis–Gerchberg algorithm for restoring lost samples. In: *Proceedings of the fifth IEEE international symposium on signal processing and information technology*, Athens, Greece, 21 December 2005, pp. 717–721. Piscataway, NJ: IEEE.
- Jung H, Lee Y, Kim B, et al. (2007) Stereo vision-based forward obstacle detection. *International Journal of Automotive Technology* 8(4): 493–504.
- Kang J and Doh NL (2016) Full-DoF calibration of a rotating 2-D lidar with a simple plane measurement. *IEEE Transactions on Robotics* 32(5): 1245–1263.
- Kawata H, Ohya A, Yuta S, et al. (2005) Development of ultra-small lightweight optical range sensor system. In: *2005 IEEE/RSJ international conference on intelligent robots and systems (IROS 2005)*, Edmonton, Canada, 2–6 August 2005, pp. 1078–1083. Piscataway, NJ: IEEE.
- Kimoto K, Asada N, Mori T, et al. (2014) Development of small size 3D lidar. In: *IEEE international conference on robotics and automation (ICRA)*, Hong Kong, China, 31 May–7 June 2014, pp. 4620–4626. Piscataway, NJ: IEEE.
- Leonard J, How J, Teller S, et al. (2008) A perception-driven autonomous urban vehicle. *Journal of Field Robotics* 25(10): 727–774.
- MacLachlan R (2005) *Tracking moving objects from a moving vehicle using a laser scanner*. Technical Report CMU-RI-TR-05-07, Robotics Institute, Pittsburgh, PA.
- Magalhaes JP, Monteiro T, Vieira JM, et al. (2013) Papoulis–Gerchberg hybrid filter bank receiver for cognitive-/software-defined radio systems. In: *IEEE international symposium on circuits and systems (ISCAS)*, Beijing, China, 19–23 May 2013, pp. 69–72. Piscataway, NJ: IEEE.
- Matthies L and Grandjean P (1994) Stochastic performance, modeling and evaluation of obstacle detectability with imaging range sensors. *IEEE Transactions on Robotics and Automation* 10(6): 783–792.
- Neumann T, Ferrein A, Kallweit S, et al. (2014) Towards a mobile mapping robot for underground mines. In: *Proceedings of the 2014 PRASA, RobMech and AfLaT international joint symposium* (eds M Puttkammer and R Eiselen), Cape Town, South Africa, 27–28 November 2014, pp. 279–284. Cape Town, South Africa: Pattern Recognition Association of South Africa.
- Noguchi H, Handa M, Fukui R, et al. (2012) Capturing device for dense point cloud of indoor people using horizontal lidar and pan rotation of vertical lidar with mirrors. In: *IEEE/SICE international symposium on system integration (SII)*, Fukuoka, Japan, 16–18 December 2012, pp. 428–433. Piscataway, NJ: IEEE.
- Ohno K, Kawahara T and Tadokoro S (2009) Development of 3D laser scanner for measuring uniform and dense 3D shapes of static objects in dynamic environment. In: *IEEE international conference on robotics and biomimetics*, Bangkok, Thailand, 22–25 February 2009, pp. 2161–2167. Piscataway, NJ: IEEE.
- Olson E (2010) A passive solution to the sensor synchronization problem. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 18–22 October 2010, Taipei, Taiwan, pp. 1059–1064. Piscataway, NJ: IEEE.
- Özbay B, Kuzucu E, Gül M, et al. (2015) A high frequency 3D lidar with enhanced measurement density via Papoulis–Gerchberg. In: *IEEE international conference on advanced robotics*, Istanbul, Turkey, 27–31 July 2015, pp. 543–548. Piscataway, NJ: IEEE.
- Papoulis A (1975) A new algorithm in spectral analysis and band-limited extrapolation. *IEEE Transactions on Circuits and Systems* 22(9): 735–742.
- Pham TQ and Van Vliet LJ (2005) Separable bilateral filtering for fast video preprocessing. In: *IEEE international conference on multimedia and expo*, Amsterdam, Netherlands, 6 July 2005. Piscataway, NJ: IEEE.
- Ponsa D, Serrat J and López AM (2011) On-board image-based vehicle detection and tracking. *Transactions of the Institute of Measurement and Control* 33(7): 783–805.
- Premebida C, Garrote L, Asvadi A, et al. (2016) High-resolution lidar-based depth mapping using bilateral filter. In: *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*, Rio de Janeiro, Brazil, 1–4 November 2016, pp. 2469–2474. Piscataway, NJ: IEEE.
- RIEGL Laser Measurement Systems GmbH (2017) Info sheet, RIEGL VUX-1 series. Available at: riegl.com/uploads/tx_pxpriegl/downloads/Infosheet_VUX-1series_2017-12-04.pdf (accessed 18 September 2017).
- Sabatini R, Gardi A and Richardson M (2014) Lidar obstacle warning and avoidance system for unmanned aircraft. *International Journal of Mechanical, Aerospace, Industrial and Mechatronics Engineering* 8(4): 718–729.
- Salari E and Bao G (2012) Super-resolution using an enhanced Papoulis–Gerchberg algorithm. *IET Image Processing* 6(7): 959–965.
- Scanse LLC (2014) User's Manual SWEEP V1.0. Available at: https://s3.amazonaws.com/scanse/Sweep_user_manual.pdf (accessed 18 September 2017).
- Sun Y, Tao J, Chen H, et al. (2015) An iterative algorithm for computed tomography image reconstruction from limited-angle projections. *Journal of Shanghai Jiaotong University (Science)* 20(2): 202–208.
- Surmann H, Lingemann K, Nüchter A, et al. (2001) A 3D laser range finder for autonomous mobile robots. In: *Proceedings of the 32nd international symposium on robotics*, Seoul, Korea, 19–21 April 2001, vol. 19(21), pp. 153–158. Seoul: Intelligent Microsystem Center.
- Thrun S, Montemerlo M, Dahlkamp H, et al. (2006) Stanley: the robot that won the DARPA Grand Challenge. *Journal of Field Robotics* 23(9): 661–692.
- Travis W, Simmons AT and Bevlly DM (2005) Corridor navigation with a LiDAR/INS Kalman filter solution. In: *Proceedings of the IEEE intelligent vehicles symposium*, Las Vegas, NV, USA, 6–8 June 2005, pp. 343–348. Piscataway, NJ: IEEE.
- Velodyne LiDAR, Inc. (2012) User's manual and programming guide HDL-64E S2 and S2. 1 high definition LiDAR sensor. Available at: www.velodynelidar.com/lidar/products/manual/63-HDL64E%

- 20S2%20Manual_Rev%20D_2011_web.pdf (accessed 18 September 2017).
- Velodyne LiDAR, Inc. (2016a) User's manual and programming guide HDL-32E high definition LiDAR sensor. velodynelidar.com/lidar/products/manual/63-9113%20HDL-32E%20manual_Rev%20E_NOV2012.pdf (accessed 18 September 2017).
- Velodyne LiDAR, Inc. (2016b) User's manual and programming guide VLP-16 Velodyne LiDAR Puck. velodynelidar.com/docs/manuals/VLP-16%20User%20Manual%20and%20Programming%20Guide%2063-9243%20Rev%20A.pdf (accessed 18 September 2017).
- Weiss B (2006) Fast median and bilateral filtering. *ACM Transactions on Graphics* 25(3): 519–526.
- Wijesoma WS, Kodagoda KS and Balasuriya AP (2004) Road-boundary detection and tracking using ladar sensing. *IEEE Transactions on Robotics and Automation* 20(3): 456–464.
- Wulf O and Wagner B (2003) Fast 3D scanning methods for laser measurement systems. In: *14th international conference on control systems and computer science*, Bucharest, Romania, 2–5 July 2003, pp. 2–5. Bucharest, Romania: Editura Politehnica Press
- Xu W and Chamzas C (1983) On the extrapolation of band-limited functions with energy constraints. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 31(5): 1222–1234.
- Yu X and Yu H (2011) A novel low-altitude reconnaissance strategy for smart UAVs: active perception and chaotic navigation. *Transactions of the Institute of Measurement and Control* 33(5): 610–630.