A mapping method in uneven terrain based on hand-held laser scanning and PDR

Xin Li 1,2*, Jian Wang 1
1. School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China
2. School of Environmental Science and Spatial Informatics, China University of Mining and Technology, Xuzhou, China

Abstract: This paper proposes a scheme for mapping in uneven outdoor and undermine environment by combining the use of hand-held laser scanning and inertial measurement unit (IMU). In order to deal with the uneven road surface, a plane mapping algorithm on the laser data was provided. Then two kinds of odometry methods were analysed, one is based on laser scanning and IMU fusion by extended Kalman filter (EKF), the other one is an improved pedestrian dead reckoning (PDR) algorithm based upon multi-threshold step detection and hybrid heading estimation. Two experiments were performed, one is besides the first floor of the School of Environmental Science and Spatial Informatics (SESSI) building on the campus of China University of Mining and Technology (CUMT) campus, another is under a gypsum mine located in Shandong Province of China. The results confirm that the PDR algorithm was better in uneven environment to produce the odometry which was used as the input of a map building tool named Gmapping. The proposed scheme can reliably achieve the mapping work in both environments.

Keywords: uneven terrain; pedestrian dead reckoning; EKF; outdoor mapping; mine mapping

1. Introduction

Building maps of the explored environment is important in order to locate the information acquired through robot sensors. A lot of work has been done on mapping 2D large environment based on simultaneous localization and mapping (SLAM) technology. In essence, the SLAM problem is one of determining the shape of the environment from local sensors and odometry data, while the same maintaining an estimate of the measuring vehicle’s relative location and orientation.

This paper proposes a scheme for mapping in outdoor and undermine environment by combining the use of hand-held laser scanning and inertial measurement unit (IMU). Outdoor mapping has been pursued by various research groups around the world. As for mine mapping, Corke and colleagues[4] have built vehicles that can acquire and utilize accurate 2-D maps of flat mines. Similarly, Baily[2] reports 2-D mapping results of an underground area using advanced mapping techniques. In [10], two line scanners were integrated on an dump truck. The iterative closest point (ICP) algorithm was used to register the 2D profiles to an existing map. This implementation was extended to mine mapping in [11]. Thrun[18,19] produced 2D maps and partial 3D models of tunnels, using a SLAM approach with two line scanning lasers mounted on a teleoperated robot. In April 2011, researchers at CSIRO[22] developed a 3D SLAM solution consisting of a spinning 2D lidar and industrial-grade IMU which was customized for mapping the Northparkes Mine in New South Wales, Australia.

However the mapping system mentioned above all make the common (but unrealistic) assumption of a flat floor, and does not address the errors introduced in the scan matching because of varying bank and elevation angles. These systems have been used to generate accurate 2D or 3D maps of relatively flat floors. Meanwhile, these systems have been designed to map that are accessible to a ground robot.

To accommodate uneven terrain, the robot platform was not rugged enough to cross stairs in outdoor environment, which can be seen in figure 1. As for the mine, there are frequently a large number of slopes, stones on the tunnel, like in figure 2. So the lightweight and portable hand held laser scanner is more suitable for the uneven environment. Walking in rugged terrain with a hand-held laser is difficult to keep horizontal, as a result, objects can appear and disappear erratically from the sensor field of view, and persisting objects can appear to change shape or shift location. So the laser scans should be projected onto the xy-plane before used.

Figure 1. Stairs in outdoor environment.

Figure 2. Stones under the mine.

This article presents a mapping method using a hand-held laser and IMU. SLAM algorithm based on particle filter [7,16] is used to generate the two-dimensional planar graph. The remainder of the paper is organized as follows: Section 2 proposes an approach that
estimates an orthogonal projection of the laser scan data, and in Section 3, a dynamic mapping algorithm based on particle filter is demonstrated. Subsequently, in Section 4, in order to improve the accuracy of the proposal distribution in particle filtering algorithm, two methods are compared and then the PDR algorithm [9, 21] is chosen for odometry calculation. Finally, experiments are analyzed in Section 5, and Section 6 concludes the paper.

The specific course is shown in Figure 3 below:

2. Laser Projection

So we project the laser scans onto the xy-plane, to make them independent of the roll and pitch motion of the vehicle. An IMU attached on the laser provides reasonably accurate information about the heading, elevation and bank angles.

The projection step is as follows: from a set of points \( S \) in the world reference frame in Cartesian coordinates, we can obtain an orthogonally projected set \( S^* \) by discarding the z value of each point in \( S \). Then, we can perform scan matching on two consecutive \( S^* \) sets. The following paragraph uses symbols from [10], for a laser scanner \( L \), positioned at \( [L]_{\text{world}} \) in the world frame, \( [L]_{\text{laser}} \) in its own frame, and \( [L]_{\text{ortho}} \) in the orthogonal reference frames. The transformation between the world, laser, and orthogonal reference frames is given by:

\[
[L]_{\text{laser}} = T_{w \rightarrow o}[L]_{\text{world}}
\]

\[
[L]_{\text{ortho}} = T_{w \rightarrow o}[L]_{\text{world}}
\]

Where

\[
[L]_{\text{world}} = [L_x, L_y, L_z, L_f, L_{\phi}, L_{\theta}]
\]

\[
[L]_{\text{laser}} = [0, 0, 0, 0, 0, 0]
\]

\[
[L]_{\text{ortho}} = [L_x, L_y, 0, 0, 0, L_{\theta}]
\]

We get the \( L_f, L_{\phi}, L_{\theta} \) from imu. The relationship between the laser, orthogonal, and world frame is shown in Fig. 4.

The laser reading from \( L \) is in the form \( [r_i, \theta_i] \), where \( r_i \) are range readings, and \( \theta_i \) are angles at which the readings were taken. The points detected by the laser thus have coordinates.
\[ P_i^{\text{laser}} = [r, \cos \theta_i, r, \sin \theta_i, 0]^T \]  
\[ T_{o-s} = (T_{w-o})^{-1} T_{w-s} \]  

Algorithm 1 gives the process of the project of points set \( S \):

begin
  for each \( i \) in \( S \) do
    if \( r_i > r_{\text{min}} \) then
      \( \theta_i = \text{angle} \_ \text{min} + i \_ \text{angle} \_ \text{increment} \)
      \[ P_i^{\text{laser}} = [r, \cos \theta_i, r, \sin \theta_i, 0]^T \]
      \[ P_i^{\text{ortho}} = T_{o-s} P_i^{\text{laser}} \]
      add \( P_i^{\text{ortho}} \) to \( S^* \)
    end
  end
  return \( S^* \)
end

If the range reading \( r_i \) is valid, then the corresponding \( \theta_i \) is calculated by the angle_increment parameter, which is 0.25\(^\circ\) for HOKUYO UTM-30LX-EW. Then the transformation between the laser and orthogonal reference frames is implemented according to formula (4) and finally the transformed points are added to the orthogonally projected set \( S^* \). The result of the orthogonal projection process is shown in Fig. 5.

3. Mapping Based Particle Filter

We use the mapping tools named Gmapping which is provided by Robot Operating System (ROS) [15]. Gmapping is a laser-based SLAM algorithm as described by [6]. Furthermore, it is the most widely used SLAM package in robots worldwide. This algorithm is a Rao-Blackwellized particle filter SLAM approach. Each particle has its own map and robot pose. The implementation works as follows: each time a new pair odometry/laser reading is received, the particle's robot pose is updated according to the motion model. This pose is subsequently used for initializing a scan matching algorithm. The scan matcher performs a local optimization for each particle. It is initialized with the pose drawn from the motion model, and the pose is correct according to each particle map. In order to avoid unnecessary computation the filter state is updated only when the robot moves more than a given threshold.

The key idea of the Rao-Blackwellized particle filter for SLAM is to estimate the joint posterior the \( P(x_{1:T}, m, z_{1:T}, u_{1:T-1}) \) from and the trajectory \( X_{1:T} = x_1, \ldots, x_T \) of the robot. This estimation is performed given the observations \( z_{1:T} = z_1, \ldots, z_T \) and the odometry measurements \( u_{1:T-1} = u_1, \ldots, u_{T-1} \) obtained by the mobile robot. The Rao-Blackwellized particle filter for SLAM makes use of the following factorization

\[ P(x_{1:T}, m, z_{1:T}, u_{1:T-1}) = P(m) P(x_{1:T}, z_{1:T}, u_{1:T-1}) \]  

This factorization first estimate only the trajectory of the robot and then to compute the map given that trajectory. Since the map strongly depends on the pose estimate of the robot, this approach offers an efficient computation. Typically, Eq. (5) can be calculated efficiently since the posterior over maps \( P(m|x_{1:T}, z_{1:T}) \) can be computed analytically using “mapping with known poses” (Moravec, 1998) since \( x_{1:T} \) and \( z_{1:T} \) are known.

To estimate the posterior \( P(x_{1:T}, z_{1:T}, u_{1:T-1}) \) over the potential trajectories, one can apply a particle filter. Each particle represents a potential trajectory of the robot. Furthermore, an individual map is associated with each sample. The maps are built from the observations and the trajectory represented by the corresponding particle.

One of the most common particle filtering algorithms is the sampling importance resampling (SIR) filter. A Rao-Blackwellized SIR filter for mapping incrementally processes the sensor observations and the odometry readings as they are available. It updates the set of samples that represents the posterior about the map and the trajectory of the vehicle. The process can be summarized by the following four steps:

1) Sampling: The next generation of particles \( x_{1:T}^{(i)} \) is obtained from the generation \( x_{1:T}^{(i-1)} \) by sampling from the proposal distribution \( \pi \). Often, a probabilistic odometry motion model is used as the proposal distribution \( \pi \). That is for all particles, the pose of next time is estimated by the motion mode, which is illustrated by algorithm 2. In reality, object motion is subject to noise. The parameters \( \lambda_1 \) and \( \lambda_2 \) are robot-specific error parameters. They model the accuracy of the motion object. The less accurate a motion object, the larger these
parameters. Here \( \lambda_3 \) and \( \lambda_4 \) are additional robot-specific parameters that determine the variance of the additional rotational noise. Finally \( x_{i+1} \) is the returned pose of next time.

**Algorithm 2:** motion model odometry \((x_{i+1}, x_i)\)

Begin
\[
\Delta = x_{i+1} - x_i = (\Delta_x, \Delta_y, \Delta_y, \Delta_z, \Delta_y, \Delta_y, \Delta_z, \Delta_y, \Delta_y)
\]
\[
x_{\text{noise}} = \text{gauss}(\lambda_3 \Delta_x + \lambda_4 \theta + \lambda_2 \Delta_x)
\]
\[
y_{\text{noise}} = \text{gauss}(\lambda_3 \Delta_y + \lambda_4 \theta + \lambda_2 \Delta_y)
\]
\[
\theta_{\text{noise}} = \text{gauss}(\lambda_3 \theta + \lambda_4 \sqrt{\Delta^2_x + \Delta^2_y})
\]
return \( x_{i+1} = x_i + p_{\text{noise}} \)

end

2) Importance Weighting: An individual importance weight \( W_{i}^{(i)} \) is assigned to each particle according to the importance sampling principle \( W_{i}^{(i)} \). The weights account for the fact that the proposal distribution \( \pi \) is in general not equal to the target distribution of successor states.

\[
W_{i}^{(i)} = \frac{P(x_{i+1}^{(i)} | z_{i+1}, u_{i+1})}{P(x_{i}^{(i)} | z_{i}, u_{i-1})} \tag{6}
\]

3) Resampling: Particles are drawn with replacement proportional to their weight. This step is necessary since only a finite number of particles is used to approximate a continuous distribution. Furthermore, re-sampling allows us to apply a particle filter in situations in which the target distribution differs from the proposal. After resampling, all the particles have the same weight.

4) Map Estimation: For each particle, the corresponding map estimate is computed based on the trajectory \( X_{1:t}^{(i)} \) of that sample and the history of observations \( z_{1:t} \).

4. Two Odometry Algorithm for Mapping

* It leaves open how the proposal distribution should be computed and when the resampling step should be carried out. One needs to draw samples from a proposal distribution \( \pi \) in the prediction step in order to obtain the next generation of particles. Intuitively, the better the proposal distribution approximates the target distribution, the better is the performance of the filter.

* As a result, typical particle filter applications [14,23] use the odometry motion model as the proposal distribution. Better proposal distribution decreases the uncertainty about the robot's pose in the prediction step of the particle filter. As a consequence, the accuracy of the odometry has significant influence on the filtering results. The following paragraphs discuss two kinds of odometry algorithm, which can be used to obtain an estimate of the horizontal displacement of the vehicle. This estimation is then as the input for Gmapping.

4.1. Odometry Based on Laser Scan and imu

* For hand-held laser scanner, we could not get the odometry like wheel platform, otherwise using scan matching to calculate the pose of the walking man. The CSM(Canonical Scan Matcher)[3] and PSM(Polar Scan Matcher)[5] are two tools in ROS for calculate odometry by laser scan. Laser scan fusing IMU by the robot_localization package[20] in the ROS is an improved method for producing odometry. The robot_localization package provides nonlinear state estimation through sensor fusion of an arbitrary number of sensors, tracking the 15-dimensional state of the vehicle, that is the pose of the vehicle, the respective velocities and linear acceleration, defined as

\[
S = [x, y, z, roll, pitch, yaw, v_x, v_y, v_z, v_{roll}, v_{pitch}, v_{yaw}, a_x, a_y, a_z]
\]

It has an implementation of an EKF, using a 3D vehicle model to project the state forward in time, and corrects that projected estimate using perceived sensor data. Because the laser scanner is operating in a planar environment, we set the two_d_mode parameter to true. This will automatically zero out all 3D pose variables, such as \( z, roll, pitch \), their respective velocities.

* We use one IMU and one odometry information produced by laser scan. Each state estimation node in robot_localization begins estimating the vehicle's state as soon as it receives a single measurement. If there is a holiday in the sensor data (i.e., a long period in which no data is received), the filter will continue to estimate the robot's state via an internal motion model.

For odometry, only \( x, y \) and \( v_x, v_y \) are used, \( yaw \) and \( v_{yaw} \) are not fused because the large bias from the ground truth. As for \( v_y \), although the walking man can't move instantaneously sideways, however, if the odometry message reports a zero value for \( v_y \) velocity, it's best to feed that value to the filter. A zero measurement in this case indicates that the walking man cannot ever move in that direction, it serves as a perfectly valid measurement.

For IMU, \( yaw, v_{yaw}, a_x \) are fused. Unlike \( v_y \) velocity in the odometry data, we leave \( v_y \) acceleration set to false, because the accelerometer, which is noisy, will likely not produce zero values for it.

The filtering process is divided into three parts:

Prediction:
\[ X(k + 1) = F(X(k)) + W(k) \]  
\[ X(k + 1) = F(X(k)) + W(k) \]  

which represents the status variable and the system noise at time \( k \). \( F(X(k)) \) are the system transition functions, which are described in the appendix A.

Measurement:
\[ Z(k) = h(X(k)) + V(k) \]

\( h(X(k)) \) is the kth recursive filter observation function, and \( V(k) \) is the measurement noise vector.

Update: The estimates of the state vector from the extended Kalman filter can be obtained by performing a time update and a measurement update at a given instant of time:
\[ X_k^- = X_{k-1} + G_k(Z_k - h(X_k)) \]
\[ X_k^- = X_{k-1} + G_k(Z_k - h(X_k)) \]

\[ G_k = P_{k-1}H_k^T(H_kP_{k-1}H_k^T + VRV^T)^{-1} \]
\[ G_k = P_{k-1}H_k^T(H_kP_{k-1}H_k^T + VRV^T)^{-1} \]

\[ X_k^- = \Phi_kX_{k-1} \]
\[ X_k^- = \Phi_kX_{k-1} \]

\[ P_k^- = (I - K_kH_k)P_{k-1} \]
\[ P_k^- = (I - K_kH_k)P_{k-1} \]

where \( G_k \) is the gain matrix of the extended Kalman filter at time \( k \), \( H_k \) is the kth recursive filter observation matrix, \( P_k \) is the covariance matrix of the state vector at time \( k \), \( R_k \) is the covariance matrix of the measurement noise vector at time \( k \). \( Q_k \) is the covariance matrix of the system noise at time \( k \), and the subscript \( k, k-1 \) represents the state or covariance estimate from time \( k-1 \) to time \( k \).

4.2. Odometry Based on PDR

PDR algorithms are on account of counting footsteps and the length. Besides, collecting through a gyro and magnetometer, a heading in 2-dimensional plane could be evaluated. It can be described as follows about a walking man’s position:
\[ X_{i+1} = X_i + SL \times \sin \alpha_i \]
\[ Y_{i+1} = Y_i + SL \times \cos \alpha_i \]

Where \((X,Y)\) is on behalf of a mobile user’s geographical position, \( SL \) represents step length as well as \( \alpha \) stands for user heading angle. In addition, Section 4.2.1 and section 4.2.2 demonstrate a new multi-threshold step detection algorithm and a hybrid heading estimation algorithm.

4.2.1. Multi-Threshold Step Detection

The maximum time duration of a step and the minimum and maximum changes in the acceleration magnitude during one step are frequently used as parameters in techniques for avoiding faulty step detection. The dynamic time warping (DTW) algorithm provides further improvement in step detection accuracy[1,17]. While walking with a hand held laser scanner under the uneven terrain mine, the step frequency or gait is tend to change irregularly. A new multi-threshold step detection algorithm based on raw acceleration measurements is provided to reduce the false gait recognition.

Owing to the effect of sensor precision and the slightly trembling hands of the walking people, pseudo peaks and valleys of vertical acceleration can be figured out frequently. As green box in Figure 6 indicating the values of the pseudo peak and valley are much less than those of the real peak and valley, although both of them reach their max acceleration magnitude. Thus, discovering some pseudo peaks and valleys by presetting an acceleration threshold is suitable. At the time, Figure 6 illustrates that the time interval between two real peaks or two real valleys is comparatively uniform. Similarly, the time interval between the real peak and the real valley is also even, which is approximately half time interval between two real peaks or two real valleys. Nevertheless, in the majority of cases, time interval between the real peak and the pseudo peak or between the real valley and the pseudo valley is exceedingly small. Under this situation, the pseudo peaks and valleys might be further excluded by presetting a time threshold. In terms of the analysis above, the two restricted parameters groups as follows, namely, \((\Delta_p, \Delta P)\) and \((\Delta_v, \Delta V)\) are put forward to detect the real peaks and valleys.
Figure. 6 Faulty step detection induced by a pseudo-peak and a pseudo-valley.

To make real-time gait detection according to the data, this paper puts forward multi-threshold step detection model on the basis of a peak-valley detection, where the following two groups of constraint conditions can be adopted to define the peak-valley detection constraints with:

1) $a_p$ and $a_v$ separately represent the amplitude at the top of the peak and valley on the waveform.

2) $\Delta f_p$ and $\Delta f_v$ stand for time difference between two contiguous peaks and between two adjacent valleys, respectively.

3) $\Delta f_{pv}$ and $\Delta f_{vp}$ represent respectively the time difference between the contiguous peak and valley or between the adjacent valley and peak.

Estimating the motion state of the pedestrian, like having rest or moving seen from $a_p$ and $a_v$, both of which accelerate the amplitude at the position of an extremum of the waveform. A static state indicates gait recognition is over. However, a dynamic state will trigger a further prove on whether the value is true (peak/valley) in a gait cycle. For instance, in term of peak detection, a dual time threshold $(\Delta f_p, \Delta f_{vp})$ is to be set in the light of the cyclicity of a complete gait cycle. As a result, the peak detection model can be demonstrated as:

$$\text{peak} = \begin{cases} 1, & a_p \geq \delta_v, \quad \Delta f_p \geq \delta_{\Delta f}, \quad \Delta f_{vp} \geq \delta_{\Delta f_{vp}} \\ 0, & a_p < \delta_v, \quad \|\Delta f_p\| \leq \delta_{\Delta f_p}, \quad \|\Delta f_{vp}\| \leq \delta_{\Delta f_{vp}} \end{cases}$$  \hspace{1cm} (15)

Where $\text{peak} = 1$ demonstrates that the acceleration data is the peak, $\text{peak} = 0$ illustrates that it’s not a peak and $(\delta_v, \delta_{\Delta f}, \delta_{\Delta f_{vp}})$ stands for the threshold set of a peak detection. Generally, they’re the empirical values. The experimental tests show that $\delta_{\Delta f_{vp}} = \frac{1}{2} \delta_{\Delta f_v}$ are obtained.

Afterwards, the valleys are detected through the condition set $(\alpha_v, \Delta f_v, \Delta f_{vp})$. The judgment on valleys won’t be made unless $Num_v, Num_{vp} = 0$. Specifically, $Num_v$ is on behalf of detection quantity on the valley while $Num_{vp}$ stands for detection quantity on the peaks. $Num_v, Num_{vp} = 0$ illustrates that the peak amounts are equal to the valley amounts in line with the peak-valley synchronization criterion.

$$\text{valley} = \begin{cases} 1, & \delta_{\Delta f_v} = \frac{1}{2} \delta_{\Delta f_v}, \alpha_v \geq \delta_v, \quad \Delta f_v \geq \delta_{\Delta f_v}, \quad \Delta f_{vp} \geq 3 \delta_{\Delta f_{vp}} \\ 0, & \delta_{\Delta f_v} = \frac{1}{2} \delta_{\Delta f_v}, \alpha_v < \delta_v, \quad \|\Delta f_v\| < \delta_{\Delta f_v}, \quad \|\Delta f_{vp}\| < \delta_{\Delta f_{vp}} \end{cases}$$  \hspace{1cm} (16)
Where \( \text{valley} = 1 \) represents that the acceleration data is the valley, \( \text{valley} = 0 \) demonstrates that it’s not a valley, \( \Delta t_{vp} \leq 3 \delta_{a} \) shows that the time difference between the current valley and the previous peak won’t be greater than three times of time threshold. \((\delta_{a}, \delta_{M}, \delta_{a'}')\) stand for the threshold set of valley detection, then the relationship can be obtained as follows,

\[
|v_{t}^{-}\rangle = 1 \text{ and }\delta_{M} = \delta_{M_{r}}.
\]

In terms of the two methods above, the presetting of acceleration threshold and time threshold, as well as most pseudo peaks and valleys in the data can be detected. However, a special condition with the existence of continuous peaks and valleys in the data still exists. In Figure 7, the circled position indicates the successive peaks, aimed at which, a comparison between two or multiple continuous peaks or valleys has been made according to their corresponding numerical values. The max peaks or valleys, which have been chosen as the real peak or valley in the gait detection, should be excluded because the numerical values of them are less than those of the pseudo peaks or valleys.

![Figure 7 Continuous peak](image)

To conclude, only two parameters, \( \delta_{a} \), the amplitude threshold and \( \delta_{M} \), the time threshold for gaits are required, if adopting this multi-threshold method of peak-valley gait detection. When the device is in a static condition or approaching a static condition, the acceleration extremes won’t be greater than 0.3 m/s². Hence, setting \( \delta_{a} = 2 \text{ m/s}^{2} \) should be applied to distinguish a person’s static state from his moving status. With regard to the time threshold for gait \( \delta_{M} \), an experiment is made as follows.

Collect two sets of data when people normally walk under the premise of horizontally holding of the devices: the first group of data is collected from the people who walks 36 steps (walk1) in a straight line in a normal way while the second group of data is gathered from people who turn a corner in 122 steps (walk2). Meanwhile, two groups of data about running are collected: the first group contains people who are trotting forward in a straight line in 45 steps (run1) while the second group includes who are running to turn a corner in 86 steps (run2). When time threshold parameters are set for different gaits, the results of the test will be:

<table>
<thead>
<tr>
<th>Gait Group</th>
<th>Within 5%</th>
<th>Absolutely Accurate</th>
<th>Greater than 5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>walk1</td>
<td>0.04–0.12</td>
<td>0.13–0.36</td>
<td>&lt;0.04 or &gt;0.36</td>
</tr>
<tr>
<td>walk2</td>
<td>0.03–0.10</td>
<td>0.11–0.28</td>
<td>&lt;0.03 or &gt;0.28</td>
</tr>
<tr>
<td>run1</td>
<td>0.03–0.13</td>
<td>0.14–0.15</td>
<td>&lt;0.03 or &gt;0.24</td>
</tr>
<tr>
<td>run2</td>
<td>0.13–0.15</td>
<td>0.14</td>
<td>&lt;0.13 or &gt;0.15</td>
</tr>
</tbody>
</table>

Table 1 illustrates that setting an extremely low (data in blue) or an extremely high (data in red) time threshold of gaits is not feasible. If setting, a big error in detection will be made. When setting the threshold as 0 or 0.01 in particular, pedestrian gaits will not be detected. The reason is that Formula 16 has considered the walking mechanism of users and the setting of the constraint condition, \( \Delta t_{vp} \leq 3 \delta_{a} \), reveals that the time difference between the current valley and the previous peak won’t be greater than three times of the time threshold. Therefore, when an extremely low time threshold is set, the reasonable valleys would be impossible to detect, resulting in a big detection error. Moreover, because constraint conditions (such as \( \Delta t_{vp} \geq 3 \delta_{a} \)) have already been adopted in the detection formula 16 under the premise of time difference, to guarantee normal gait detection is impossible when setting an extremely high threshold. Therefore, it is a necessity to apply a suitable threshold parameter so as to reach the anticipated precision when detecting pedestrian gaits.

On the other hand, the range of time-frequency parameter in normal walking gaits is broader than that in running state, which can be seen from Figure 8 that the acceleration data is relatively stable when the moving state is stable. When the time threshold is set in a certain range (such as 0.13 and 0.28s), the accurate detection of the step frequency of the users will be promoted. However, when people are running, the movement of their bodies (hands) will result in a large number of noise data among the acceleration data. As is shown in Figure 9, the amplitudes vary greatly in the numerical values. Then, it would have difficulties in controlling gait detection when merely applying amplitude threshold parameter. Thus, more requirements are put forward on time parameters, and choosing an appropriate time threshold has become necessary for accurate gait detection. For instance, if the time threshold is not set to be around 0.14 seconds, it would not ensure an absolutely accurate detection in the state of running.
As above, in order to unify the parameters that have been applied in the pedestrian gait detection model proposed in this paper, $\delta_a$, the amplitude threshold is set to be 2m/s$^2$. In terms of $\delta_{\Delta t}$, the time threshold for gaits, it’s set to be 0.14s, which is an appropriate numerical value that is applicable to the above four moving states.

Moreover, we’ve collected 14 groups of data about the persons of different height with different body shapes moving in different modes at different speed. According to the threshold parameters selected as above, put all of these data into our simple parameter-based gait recognition model proposed in this paper for experimental detection. The result is provided in the following table.

<table>
<thead>
<tr>
<th>SN of WalkS</th>
<th>Real steps</th>
<th>Detected steps</th>
<th>Accuracy rate</th>
<th>SN of RunS</th>
<th>Real steps</th>
<th>Detected steps</th>
<th>Accuracy rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>54</td>
<td>54</td>
<td>100%</td>
<td>1</td>
<td>51</td>
<td>50</td>
<td>100%</td>
</tr>
<tr>
<td>2</td>
<td>101</td>
<td>103</td>
<td>98%</td>
<td>2</td>
<td>62</td>
<td>62</td>
<td>100%</td>
</tr>
<tr>
<td>3</td>
<td>72</td>
<td>73</td>
<td>98.6%</td>
<td>3</td>
<td>40</td>
<td>40</td>
<td>100%</td>
</tr>
<tr>
<td>4</td>
<td>56</td>
<td>57</td>
<td>98.2%</td>
<td>4</td>
<td>43</td>
<td>44</td>
<td>97.7%</td>
</tr>
<tr>
<td>5</td>
<td>122</td>
<td>122</td>
<td>100%</td>
<td>5</td>
<td>50</td>
<td>44</td>
<td>88%</td>
</tr>
<tr>
<td>6</td>
<td>316</td>
<td>317</td>
<td>99.7%</td>
<td>6</td>
<td>45</td>
<td>47</td>
<td>95.6%</td>
</tr>
<tr>
<td>7</td>
<td>248</td>
<td>247</td>
<td>99.6%</td>
<td>7</td>
<td>86</td>
<td>84</td>
<td>97.7%</td>
</tr>
</tbody>
</table>

WalkS means that people walk at normal speed, while Runs represents that people run at different speed. The accuracy rate of gait detection reveals that our simple parameter-based multi-constraint model can control the ratio in 13/14 within a range of 2 steps during gait recognition when the users are moving in multiple modes. Moreover, we’ve utilized the FIR filter provided by MATLAB to implement the low-pass filter-based gait detection method (also called as FIR filter) proposed in Literature[9] according to the peaks and valleys. Through the comparison with our method, Table 3 shows the experimental result with the application of such a FIR filter at two kinds of walking speed mentioned as above.
also achieve an average accuracy rate of up to 96.7%, which is improved by 3.2% than the correlation between the magnetometer and the gyroscope.

The results reveal that the pseudo-heading measurements recorded by the magnetometer before and after the turn are considerably smoothed by the proposed algorithm.

The comparison reveals that our method can achieve an average accuracy rate of up to 99.2% in gait recognition when the users are walking at normal speed. This is the same accuracy rate with the data obtained based on the FIR filter in gait detection. Even when the users change their speed at different speeds, our method can also achieve an average accuracy rate of up to 96.7%, which is improved by 3.2% than the data based on the FIR filter under the same mode in gait detection. The lowest accuracy is separately 88% and 72% through both of the methods in gait detection. It reveals that our simple parameter-based gait recognition model with multiple constraints and the selection of the model parameters are of strong universality and applicability.

Step length estimation is an important factor that affects the positioning accuracy and can be performed in combination with a step detection procedure. The step length is related to the acceleration and is typically given by

$$L_k^a = K \cdot \frac{a_{max}^c - a_{min}^c}{2}$$ (17)

where $L_k^a$ is the length of the $k^{th}$ step, $a_{max}^c$ and $a_{min}^c$ are the minimum and maximum amplitudes, respectively, of the acceleration. The value of the coefficient $K$ depends on the individual and can be calibrated.

### 4.2. Hybrid Heading Estimation

Heading determination is a significant component of PDR-based positioning. The heading angle $\psi$ is defined as the angle of rotation about the $x$ axis with respect to the horizon/ground, which can be estimated using a gyroscope integrated with a magnetometer. The improved heading estimation algorithm presented by Wonho Kang is applied here [8]. The fused heading angle is calculated as follows:

$$\theta_{k^j} = \alpha \theta_{j,k} + \beta \theta_{j,\alpha} + \gamma \theta_{j,\beta},$$

where $\theta_{j,k}$, $\theta_{j,\alpha}$, and $\theta_{j,\beta}$ denote the measurements acquired by the gyroscope and the magnetometer, respectively, for the $k^{th}$ step. $\theta_{j,\alpha}$ is the standard deviation of the magnetometer, and $\theta_{j,\beta}$ is the correlation between the magnetometer and the gyroscope. $\theta_{m,k}^c$ and $\theta_{g,k}^c$ denote the measurements acquired by the magnetometer and the gyroscope, respectively, for the $k^{th}$ step. $\theta_{j,\alpha}$ is the difference between $\theta_{m,k}^c$ and $\theta_{g,k}^c$. $\theta_{m,k}^c$ is the difference in the magnetometer reading between two consecutive steps $k$ and $k-1$. $\beta_j = 0.3, \gamma_j = 0.2; \beta_v = 0.4, \gamma_v = 0.6; \alpha = 0.4, \gamma = 0.6$.

Figure 10 shows the fusion results of a test in which a person walked forward five steps with an IMU held firmly on his hand, stopped for a brief period, and eventually turned around and returned on the same path. The results reveal that the pseudo-heading measurements recorded by the magnetometer before and after the turn are considerably smoothed by the proposed algorithm.

### 5. Experiment and Analysis

The experimental data used was obtained from a 2D scanning range laser (HOKUYO UTM-30LX-EW) held with hand. The laser returns a 270° planar sweep of range measurements in 0.25° intervals with a range resolution of about ±30mm. The scan rate is 40 Hz, and maximum effective range is about 30 m. An IMU(MTi-100) was stuck on the laser, which provided the 3D Orientation and acceleration. The work rate was set to 100Hz.

#### 5.1. Test One

To compare of the two odometry algorithms, an experiment was performed besides the first floor of the School of Environmental Science and Spatial Informatics (SESSI) building on the campus of China University of Mining and Technology (CUMT) in Xuzhou, Jiangsu, China (Figure 11). The max building interval space is near 30 meters which is just close the range of the laser scanner except region

<table>
<thead>
<tr>
<th>SN of WalkS</th>
<th>Real steps</th>
<th>Detected steps</th>
<th>Accuracy rate</th>
<th>SN of RunS</th>
<th>Real steps</th>
<th>Detected steps</th>
<th>Accuracy rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>54</td>
<td>55</td>
<td>98.1%</td>
<td>1</td>
<td>51</td>
<td>50</td>
<td>98%</td>
</tr>
<tr>
<td>2</td>
<td>101</td>
<td>103</td>
<td>98%</td>
<td>2</td>
<td>62</td>
<td>61</td>
<td>98.4%</td>
</tr>
<tr>
<td>3</td>
<td>72</td>
<td>73</td>
<td>98.6%</td>
<td>3</td>
<td>40</td>
<td>39</td>
<td>97.5%</td>
</tr>
<tr>
<td>4</td>
<td>56</td>
<td>56</td>
<td>100%</td>
<td>4</td>
<td>43</td>
<td>43</td>
<td>100%</td>
</tr>
<tr>
<td>5</td>
<td>122</td>
<td>122</td>
<td>100%</td>
<td>5</td>
<td>50</td>
<td>36</td>
<td>72%</td>
</tr>
<tr>
<td>6</td>
<td>316</td>
<td>315</td>
<td>99.7%</td>
<td>6</td>
<td>45</td>
<td>47</td>
<td>95.6%</td>
</tr>
<tr>
<td>7</td>
<td>248</td>
<td>248</td>
<td>100%</td>
<td>7</td>
<td>86</td>
<td>81</td>
<td>94.2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Gait recognition based on FIR filter data</th>
</tr>
</thead>
<tbody>
<tr>
<td>SN of WalkS</td>
<td>Real steps</td>
</tr>
<tr>
<td>1</td>
<td>54</td>
</tr>
<tr>
<td>2</td>
<td>101</td>
</tr>
<tr>
<td>3</td>
<td>72</td>
</tr>
<tr>
<td>4</td>
<td>56</td>
</tr>
<tr>
<td>5</td>
<td>122</td>
</tr>
<tr>
<td>6</td>
<td>316</td>
</tr>
<tr>
<td>7</td>
<td>248</td>
</tr>
</tbody>
</table>
A. Region A is an open space with several stairs, showed in figure 1, which prone to cause errors for odometry based laser scan. The red line shows the true trajectory walking around the middle building with a hand-held laser scanner, which is acquired from a high accuracy GPS/INS.

Figure. 11 The ground true around the building.

Figure. 12 The trajectory of CSM and PSM.

The results produced by CSM and PSM tools respectively are depicted in figure 12. The deviation is too large from ground truth due to multiple factors, such as the effect of uneven flooring and the noise of long distance in the sensor measurements etcetera, so odomery from these tools can’t be used for mapping in such a harsh environment.

Figure. 13 Kalman filter fusing based CSM and IMU.

Figure. 14 Kalman filter fusing based PSM and IMU.
Then CSM and PSM tools fused with IMU are tested. In figure 13 and 14 show the output of Gmapping with the IMU fused as odometry. The mapping results still have large deviation from the true map.

When the PDR odometry is used for Gmapping’s input, the mapping results is much closer to the true environment, which can be seen from figure 15. The red line is the trajectory of PDR algorithm, green line is the Gmapping’s trajectory output. Compared with method using in 4.1, the results demonstrate that the Gmapping based PDR algorithm achieves much higher reliability and accuracy. The ground truth contains a closed loop, which allows us to start and finish at the same place. The motion estimation generates a gap between the starting and finishing positions, which indicates the amount of drift. The measured drifts of the two methods described in 4.1 and 4.2 are compared in table 4. The gap of fusing IMU method with CSM and PSM are separately 6.96m, 3.63m, which are 4.4, 3.0 times of the PDR based method, which is only 1.52m.

<table>
<thead>
<tr>
<th>Odometry Algorithm</th>
<th>Gap (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM and IMU fusing</td>
<td>6.96</td>
</tr>
<tr>
<td>PSM and IMU fusing</td>
<td>4.63</td>
</tr>
<tr>
<td>PDR</td>
<td>1.52</td>
</tr>
</tbody>
</table>

Table 4 Gap between the initial and end position.

5.2. Test Two

The experiment was conducted in a gypsum mine in Cangshan County, Shandong Province. In order to estimate the performance of the two methods, the contour profile of four pillars and the corresponding areas are computed.

Compass and measuring tape were used to record the inflection point and the length of each segment of the contour profile, as in figure 16. The measure results are regarded as true results and can be seen in figure 18: (a), (b), (c), (d).
These pillars were also mapped both by the laser scan with IMU fusing and the laser scan with PDR technology, as can be seen in Figure 17, the results are in Figure 19.

![Image](https://via.placeholder.com/150)

**Figure 18** Measuring results with compass and tape.

The outlines of the pillar sectional drawing are checked by the AutoCAD tools and then use it to calculate the corresponding area. Comparison of the two techniques is listed in Table 5:

<table>
<thead>
<tr>
<th>Pillar Index</th>
<th>Tape and Compass(m2)</th>
<th>Laser &amp; IMU (m2)</th>
<th>Deviation</th>
<th>PDR (m2)</th>
<th>Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>33.49</td>
<td>37.29</td>
<td>11.3%</td>
<td>34.89</td>
<td>4.0%</td>
</tr>
<tr>
<td>a2</td>
<td>57.21</td>
<td>53.53</td>
<td>6.4%</td>
<td>55.25</td>
<td>3.4%</td>
</tr>
<tr>
<td>b1</td>
<td>124.58</td>
<td>118.55</td>
<td>4.8%</td>
<td>122.15</td>
<td>3.5%</td>
</tr>
<tr>
<td>b2</td>
<td>87.23</td>
<td>91.24</td>
<td>4.5%</td>
<td>89.19</td>
<td>2.2%</td>
</tr>
</tbody>
</table>

In Table 5, both of a1 and a2 are the scanning methods integrated with laser and IMU. However b1 and b2 are the laser scanning methods based on the PDR algorithm. The results demonstrate that the PDR algorithm can provide relatively accurate input in short-term time for Gmapping, which then yields a consistent map and trajectory close to the truth. The trajectory is perfect with an exact loop. The deviation of measuring area of the PDR based method is no more than 4% with manual measurement method, which is significantly better than the laser scan with IMU fusing method. With respect to the efficiency and convenience, laser scan method is more suitable for underground survey.

6. Conclusions

This paper investigated method for mapping in uneven terrain by combining the use of laser scanning and IMU. An improved pedestrian dead reckoning (PDR) algorithm based upon multi-threshold step detection and hybrid heading estimation are presented. The experimental results indicate that the odometry based PDR algorithm will generate a better map, which significantly outperforms using laser scan fused with IMU in the uneven outdoor environment or underground. It demonstrates that a precise 2D map can be achieved using Gmapping with PDR as odometry. The further development of the work will focus on designing an portable and cheap 3D laser scan based on 2D laser rotation, so as to provide a more accurate 3D map.

Acknowledgements

This research work was funded by the National Natural Science Foundation of China under Grant No.41674030.

References


Appendix A: System transition functions

\( F(X(k)) \) used in robot_localization

\[
\begin{align*}
X(k+1) &= X(k) + cy sp V_c \Delta t + (cy sp cy sr) W_c \\
Y(k+1) &= Y(k) + cy sp V_c \Delta t + (cy sp cy sr) W_c \\
Z(k+1) &= Z(k) + sp V_c \Delta t + (cy sp cy sr) W_c \\
Roll(k+1) &= Roll(k) + V_{roll} \Delta t + Q_{roll} \\
Pitch(k+1) &= Pitch(k) + V_{pitch} \Delta t + Q_{pitch} \\
Yaw(k+1) &= Yaw(k) + V_{yaw} \Delta t + Q_{yaw} \\
V_{x}(k+1) &= V_{x}(k) + a_x \Delta t + Q_{x} \\
V_{y}(k+1) &= V_{y}(k) + a_y \Delta t + Q_{y} \\
V_{z}(k+1) &= V_{z}(k) + a_z \Delta t + Q_{z} \\
a_x (k+1) &= a_x(k) + Q_{a_x} \\
a_y (k+1) &= a_y(k) + Q_{a_y} \\
a_z (k+1) &= a_z(k) + Q_{a_z} \\
V_{roll}(k+1) &= V_{roll}(k) + Q_{roll} \\
V_{pitch}(k+1) &= V_{pitch}(k) + Q_{pitch} \\
V_{yaw}(k+1) &= V_{yaw}(k) + Q_{yaw} \\
\end{align*}
\]

Here, \( cr, cp, cy \) indicate \( \cos(roll), \cos(pitch), \cos(yaw) \) respectively and \( sr, sp, sy \) indicate \( \sin(roll), \sin(pitch), \sin(yaw) \). \( Q \) is noise parameters accordingly.