NaviGlass: Indoor Localisation Using Smart Glasses

Yongtuo Zhang[1], Wen Hu[1,2], Weitao Xu[3], Hongkai Wen[4], and Chun Tung Chou[1]
[1]School of Computer Science and Engineering, University of New South Wales, Australia
[2]National ICT Australia, Australia
[3]School of Information Technology and Electrical Engineering, University of Queensland, Australia
[4]Department of Computer Science, University of Oxford, United Kingdom
{ytzhang, wenh, ctchou}@cse.unsw.edu.au[1] w.xu3@uq.edu.au[3] Hongkai.Wen@cs.ox.ac.uk[4]

Abstract

Smart glasses (e.g. Google Glasses) is a class of wearable embedded devices with both inertial sensors and camera onboard. This paper proposes a smart glass-based indoor localisation method called NaviGlass. Because of high energy consumption of vision sensors, NaviGlass uses inertial sensors predominantly and uses the camera images for correcting the drift in the position estimates due to the accumulated errors of inertial sensors. Due to limited computation resources available on smart glasses, the computation time for image matching, which is needed to correct the position estimate, is high. We propose a feature reduction method that can significantly reduce the computation time for image matching but with little compromise on accuracy. We compare our method against Travi-Navi, which is a state-of-the-art localisation system that uses both inertial and image sensors. Our evaluations show that our proposed method achieved a mean localisation error of 3.3m which is 64% less than that of Travi-Navi.

Categories and Subject Descriptors

C.5 [**COMPUTER SYSTEM IMPLEMENTATION**]: Miscellaneous; B.8.2 [**PERFORMANCE AND RELIABILITY**]: Performance Analysis and Design Aids

General Terms

Design, Performance

Keywords

Indoor localisation, Smart glasses, Feature reduction

1 Introduction

Indoor localisation is a well researched topic in the last two decades. Many methods have been proposed to provide indoor localisation services for embedded devices. These methods can be divided into two categories: infrastructure based and infrastructure free. For infrastructure based methods, localisation is realised with the help of external equipment. A well known example of infrastructure based method is WiFi fingerprinting [37, 26]. The key idea of WiFi fingerprinting is to use the signal strength of a few WiFi access points as the signature of each location. Although WiFi fingerprinting has met with some success, a major challenge it faces is that the WiFi signature is not static and changes over time. This means frequent re-training is required to ensure its feasibility. For infrastructure free method, localisation is realised with the embedded device alone with little help from external equipment. A well known example of infrastructure free method is Pedestrian Dead Reckoning (PDR). PDR makes use of Inertial Measurement Unit (IMU) sensors commonly available on embedded device. An IMU typically includes a 3-axis accelerometer, gyroscope and magnetometer. In principle, one can obtain position estimate by numerically integrating the IMU measurements. However, the reality is much more complex due to noisy measurements, un-structured way in which the subject carries the embedded device and the fact that walking is not a steady movement. These challenges drive the researchers to derive many new methods to improve the accuracy of PDR for indoor localisation, e.g. by using particle filters [19, 29], corrections by using WiFi [34, 41], landmarks [23] or indoor floor plan [26, 15]. In this paper, we propose a new localisation method which combines PDR and camera on smart glasses. To be best of our knowledge, this is the first attempt in using only smart glasses for indoor localisation.

Smart glasses, e.g. Google Glass and Vuzix M100, are a new class of wearable embedded device that are worn like sun glasses or prescription glasses. A key advantage of smart glasses is that they provide a way for a user to interact with an embedded device in a hand-free manner. Motivated by that, in this paper, we will instead focus on the use of smart glasses for indoor localisation since it can provide a better user experience than smart phones while walking. Most smart glasses available today are equipped with IMUs and camera sensors. Both of these sensors can be used for localisation, either on their own, or together. We know from the previous discussion that using IMU measurements alone for localisation by using PDR does not provide accurate localisation estimates. A common problem is that the localisation estimation error grows over time, which is referred to as

International Conference on Embedded Wireless
Systems and Networks (EWSN) 2016
15–17 February, Graz, Austria

ISBN: 978-0-9949886-0-7

drifting. Instead of using IMU sensors, one can use the image sensor on smart glasses for indoor localisation. The basic idea is to use visual features to characterise each location. By comparing the visual features at the current location against those stored on the device, one can determine the location of the device. In fact, a very successful vision-based Simultaneous Localisation and Mapping (SLAM) algorithm invented by robotics researchers is FAB-MAP [9, 10]. However, a pure vision-based localisation algorithm is not feasible for smart glasses because of limited power supply and computation resources. For example, our experience shows that some smart glasses take nearly 30 seconds to process an image with a resolution of 960×720 to provide a greater than 90% accuracy to match an image to those in the database. The resource limitation can severely limit the sampling rate of the camera and the resulting localisation estimate is less than desirable.

Since neither pure PDR-based nor pure vision-based localisation provides good localisation estimate, this paper proposes a new localisation method which fuses PDR and vision. We call our proposed method NaviGlass. Due to high resource consumption of vision-based component, NaviGlass uses PDR predominantly and uses vision to correct the drift of position estimate in PDR. The key challenge in realising NaviGlass is to reduce the computation time needed to process an image. In particular, this paper makes the following contributions:

- The time to match an image to those in the database increases with the number of visual features being used. In order to reduce the time required to perform matching, we propose a novel feature reduction method which identifies the informative visual features. The method makes use of the concept of mutual information to discard uninformative features or to merge features. We will show that this method is effective in reducing the number of features and consequently the time to do matching.
- We implemented our NaviGlass on real smart glasses and compared NaviGlass against Travi-Navi, which is a state-of-the-art indoor localisation method that uses both PDR and vision. Our results show that NaviGlass has a better image matching accuracy compared to Travi-Navi. Also, NaviGlass achieves a mean localisation error of 3.3m, which is 64% less than that of Travi-Navi.

The rest of the paper is organised as follows. Sections 2 and 3 review the related work and technical background. Section 4 describes NaviGlass. Performance evaluation of NaviGlass is given in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

Many indoor localisation methods have been proposed to localise mobile and embedded devices. Given this paper uses both PDR and vision-based localisation methods, the discussion here will be predominantly focused on these two classes of methods.

PDR uses IMU — accelerometer, gyroscope and magnetometer — to estimate the position of the subject [19, 34]. The concept behind PDR is pretty simple. Since acceleration is the second derivative of position, one can obtain position from acceleration using numerical integration. However, there are many barriers to achieving accurate indoor localisation with PDR using IMU measurements from embedded devices. First, the IMU measurements from embedded device are a lot more noisier than those high end IMU units for aviation [34]. As a result, a naïve numerical integration will result in accumulation of measurement errors. Researchers have tried to counter measurement noise by using Kalman filters and particle filters [15, 19]. Second, people carry embedded devices in many different ways: in their hands, pockets, hand bags etc. [16]. Therefore, researchers need to derive methods to rotate the measurements to the movement axis. Third, the movement of human is inherently jerky, rather than steady. This drives researchers to provide accurate step detection and counting [19, 30]. Fourth, magnetic distortion exists in an indoor environment due to the use of magnetic materials (e.g. iron, steel) in the construction of modern buildings [7, 14]. The consequence is that heading estimation based on indoor magnetometer measurements can deviate from the correct heading by up to 180°. Such magnetic distortion has to be corrected. All these issues cause the PDR location estimate to be highly erroneous. Many solutions have been proposed to address these issues. Some methods rely on IMU sensing alone and use one type of measurements to correct the error in another type of measurements [19]. The majority of PDR methods use external knowledge to correct location estimations . The external knowledge that researchers have investigated include using floor plan [15], landmarks[23], WiFi signal strength [5, 20] and access points [35], earth's nominal magnetic field [7], vision [41].

Vision-based localisation is also a well-studied research area. FAB-MAP [9, 10] is a successful vision-based SLAM algorithm in robotics based on recursive Bayes estimation; we will explain FAB-MAP in greater details in Section 3. The accuracy of vision-based localisation methods critically depends on the accuracy of matching the image at the current location to those in the database. An accurate image matching method is based on the Bag of Words (BoW) concept which has been applied in many vision-based localisation systems [17, 18, 28]. More sophisticated vision methods have also been applied, for example 3D model [40] and panoramic images [31]. Due to the extensive computation needed for image matching, either the platform must be equipped with significant computing resources or the computation to be carried out on a remote server [18]. Unfortunately neither of these solutions are feasible for embedded platform because of limited computational resources available on board or limited bandwidth available on board to transfer images. Instead in this paper, we investigate the use of feature reduction method to reduce the computation requirements on embedded platform.

With the development of smart glasses, many new applications have been proposed, e.g. social contact [38], customer analytics [27], cognitive assistance [13], human recognition [33] and others. These applications make extensive use of camera and sensors. The challenge of limited computation resources also applies to these applications on off-the-shelf smart glasses. The feature reduction methodology proposed in this paper can potentially be applied to these applications as well.

This paper makes use of visual information to correct PDR location estimate. Two earlier examples of such method can be found in [41, 2]. In particular, we make a comparison with Travi-Navi [41] and our evaluation shows that our proposed method outperforms Travi-Navi.

The computation bottleneck in our system is due to the long time to match an image to those in the database in order to determine the subject's location. The time required for matching is an increasing function of the number of features. A method to address this bottleneck is to use informative features for matching. Feature selection and feature reduction have been widely investigated in the past. A well-known feature selection method is Principal Components Analysis (PCA) [22] but it is difficult to apply such methods to binary features that are used FAB-MAP. An alternative approach is to apply mutual information to select good features [36, 25, 12]. This is based on the information theoretic criterion that the best set of features is the one that maximises its mutual information with the decision classes [12]. Our proposed feature selection method is based on the same criterion but we introduce the concept of feature merging which does not seem to have been used before.

Since a smart phone has an IMU and an on-board camera, it is in principle possible to implement a localisation method on a smart phone using IMU and vision inputs. However, we would like to argue against this idea from a practical point of view. In order to correct the localisation estimate, the smart phone has to be carried in such way that its camera is facing in the direction that the user is walking. There does not appear to be a practical way for a user to do that. It is rather awkward to ask a user to hold the smart phone by hand so that its camera is facing the direction that the user is walking. Although a user can carry the phone on a lanyard but the camera is likely to produce blurred images, which are not useful at all. We therefore see smart glasses as a better way to carry an IMU sensor and a camera, compared to smart phones.

3 Technical Background

Our proposed indoor localisation method NaviGlass is based on FAB-MAP, which is a state-of-the-art vision-based SLAM algorithm for robots working in the outdoor environment. This section provides a technical overview of FAB-MAP to make this paper self-contained. For complete description of FAB-MAP, see [9, 10].

FAB-MAP is an algorithm derived primarily for loop-closure detection, which means detecting whether the robot has returned to a place that it has been before, in a large unknown environment. FAB-MAP realises loop-closure by using the appearance at each location. When a robot which is equipped with FAB-MAP moves around, it takes pictures of the environment and extracts visual features from the pictures. To the robot, each location is characterised by the visual features present in the pictures taken at that location. When the robot moves to a different location, it wants to tell whether it has moved to a place that it has never been before or a place that has been previously visited. The robot does that by taking pictures of the current location and extracting visual features. It then compares the visual features of the current location against those of the previous places that it has visited. If a match occurs, then the robot knows that it has returned to a previously visited location; otherwise, the robot decides that it has gone to a new location. In this paper, we will use the term **image matching** to refer to the comparison of visual features at the current location against the stored features from earlier visited locations.

FAB-MAP is a highly accurate SLAM algorithm for a couple of reasons. First, it uses robust visual features such as Speeded Up Robust Feature (SURF) [3]. Second, FAB-MAP uses a Bayesian method to compute the posteriori probability for image matching. It is a fine-grained method requiring the probability distribution of the visual features at various location. We will now give further description FAB-MAP which are important for understanding the rest of this paper.

When FAB-MAP captures an image, it determines the visual features in the image. In particular, we will assume that SURF features are used. SURF is a popular method to extract visual features from images. In particular, it extracts features which are scale invariant. This makes SURF features robust because useful visual features should be present in all the spatial scales. SURF features can be extracted by a Hessian detector which computes the Hessian at a number of spatial scales. The number of SURF features in an image can be controlled by an Hessian threshold.

FAB-MAP uses the Bag of Words (BoW) framework to represent the visual features. Assuming that the robot has been to a number of locations and has taken pictures at these locations. The robot has also extracted visual features of these locations. Within the BoW framework, each visual feature is considered as a *visual word* and each location is characterised by a set of of visual words found at that location. The collection of all the visual words from all the locations is known as the vocabulary. Let $\mathcal{V}$ denote the vocabulary which contains $|\mathcal{V}|$ visual words. The robot takes an image at time k and extracts the visual words (= visual features) from the image. This image taken at time k can be characterised by the $|\mathcal{V}|$-dimensional vector $Z_k = \{z_1, z_2, ..., z_{|\mathcal{V}|}\}$ where z_j is a binary variable indicating whether the j-th visual word is observed in the image taken at time k. (The vector Z_k will be referred to as a BoW vector and the collection of BoW vectors at a number of locations form a BoW matrix.) At time k, the robot wants to know whether it is located at a place L_i that it has visited before. It does that by estimating the posteriori probability that it is at location L_i given all the past observations $\mathcal{Z}_{0:k}$ available up to time k. By using recursive Bayesian estimation, we have

$$p(L_i|\mathcal{Z}_{0:k}) = \frac{p(Z_k|L_i, \mathcal{Z}_{0:k-1})p(L_i|\mathcal{Z}_{0:k-1})}{p(Z_k|\mathcal{Z}_{0:k-1})} \tag{1}$$

It is difficult to calculate the right-hand side of Eq. 1 exactly because it requires probability distribution of very high dimension. To simplify calculations, FAB-MAP assumes that the current observation Z_k at time k is independent of the past

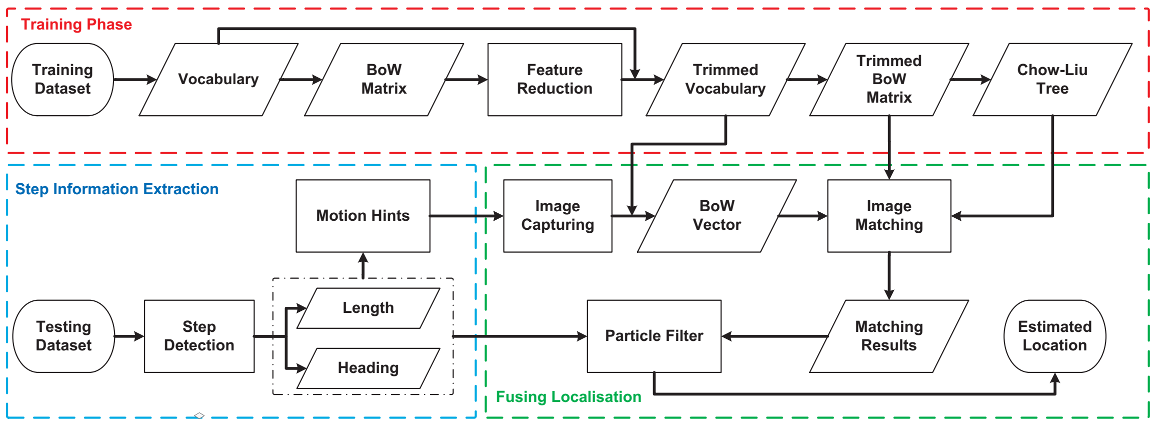


Figure 1: The structure of our proposed indoor localisation method NaviGlass.

observations $Z_{0:k-1}$, i.e. it assumes that

$$p(Z_k|L_i, Z_{0:k-1}) \approx p(Z_k|L_i) \tag{2}$$

With this simplification, the goal is now to estimate $p(Z_k|L_i)$. This probability calculation is also difficult because it requires the joint probability distribution of z_1, z_2, ... , $z_{|\mathcal{V}|}$ at location L_i, which is a probability distribution of very high dimension because of the number of visual words $|\mathcal{V}|$ in the vocabulary is large. FAB-MAP simplifies this calculation by using Chow-Liu Tree [6] which is a method of approximating a high dimension joint probability distribution and is optimal in some information theoretic criterion. A Chow-Liu Tree can be derived by using only the joint probability distribution of each pair random variables, which can be estimated empirically using only second order statistics. With Chow-Liu Tree, the probability $p(Z_k|L_i)$ can be computed using first and second order probability on z_j.

FAB-MAP is a popular SLAM method among robotics researchers because it gives good accuracy. However, FAB-MAP requires extensive computation in order to determine the location. This can be a serious issue in indoor localisation because most embedded devices have only limited computation power. In the next section, we will explain how we can adapt FAB-MAP to work in a resource constraint embedded platform to carry out accurate indoor localisation.

4 NaviGlass

This section presents an overview of our proposed indoor localisation method NaviGlass on smart glasses. Figure 1 shows a block diagram depicting the data processing components in NaviGlass.

4.1 Overview of NaviGlass

Figure 1 shows that NaviGlass makes use of both IMU and camera. The IMU is for PDR. The image is used to correct the inaccuracies of PDR by matching an image to a location, based on the FAB-MAP framework. The method begins with off-line training to obtain a low-complexity description of the image features at each location. The online operations consist of a number of components: step detection, image taking, PDR and data fusion.

We assume that the smart glasses will be used for localisation within a particular target area in which the off-line training takes place. During the training, a subject walks around the area with smart glasses on. The smart glasses take picture of the environment at a uniform sampling rate. At the end of the walk, the pictures are transferred to an off-line server for processing. The pictures are first tagged with the correct location. Features are then extracted from the images. The vocabulary consists of all the features found among the training image set. Following FAB-MAP, a BoW matrix can be computed based on the vocabulary. The role of the BoW matrix is to match an image taken during the online operation to a location in the target area. The size of this BoW matrix is large due to the large number of features. The number of features has a significant impact on the amount of time it takes to process and match an image during the online operation of the localisation algorithm. The time required is exacerbated by the limited computation resources available on the glasses. In Section 4.2, we present a feature reduction method which reduces the number of features while maintaining a good image matching accuracy for our type of image matching problem. We then use the reduced BoW matrix to compute a Chow-Liu tree for image matching based on the FAB-MAP framework. Both the reduced BoW matrix and Chow-Liu Tree will be stored on the glasses for online localisation.

Since IMUs have lower energy consumption, our localisation is primarily based on PDR. For PDR, we use accelerometer readings to detect the steps and the gyroscope to determine the direction of movement. In order to cope with the noise in the IMUs, we use a particle filter to produce a position estimate. It is well know that the localisation error of PDR grows over time (drifting). Our proposal is to use camera images to correct this drift where image matching is to be performed in the FAB-MAP framework. Since the accuracy of image matching depends strongly on the quality of image taken by the camera [41]. Our system avoids taking images when the user is walking at high acceleration in order to reduce the chance of getting a blurred image. We describe our step detection method and image taking method in Section 4.3. Since image matching does not have 100% accuracy, it

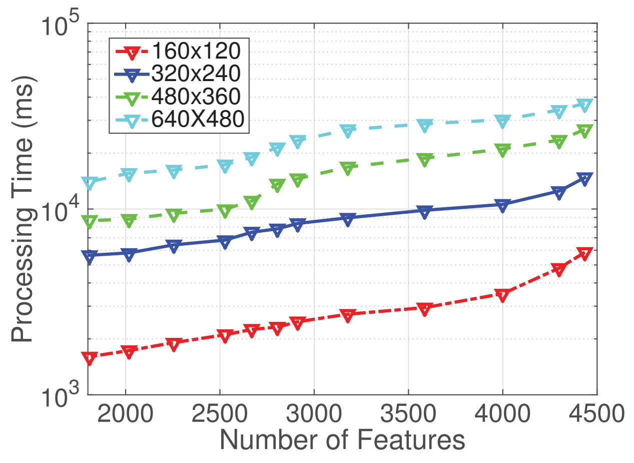


Figure 2: Time consumption when processing images using different number of features and resolutions on Vuzix M100 Glass

is possible for an incorrect image matching result to produce highly incorrect position estimate. We therefore put in additional safe guard to detect an occasional incorrect image matching. The location of the image matching result is then incorporated (fused) into the particle filter to correct the position estimate. We will describe our image matching method and our data fusion method in Section 4.4. Note that our proposed method does not use the floor plan of the target area as prior information though such information can be easily introduced into the particle filter.

4.2 Feature Reduction Methodology

Figure 2 shows that there is a positive correlation between the number of features and computation time. However, standard feature reduction algorithms based on PCA [22] and mutual information [25] cannot be applied to our type of features. Hence, we propose a customised off-line feature reduction algorithm that can significantly reduce the computation time on commodity smart glasses for image matching but with little compromise on accuracy.

We start by setting up the feature reduction problem. We assume that we have taken m images at m different locations at the training phase. (Note that for simplicity, we are assuming that there is an image for each location. The feature reduction method can also be applied when there are multiple images for each location.) Our aim is to use the visual features in these m images to identify the m different locations. We assume that initially each image is characterised by n binary features. We will use i to index the images, and q and r to index the features. We use the number $z_{i,q} \in \{0,1\}$ to indicate whether the q-th feature is present or absent in the i-th image. If $z_{i,q} = 1$, then it means that the q-th feature is present in the i-th image; otherwise, $z_{i,q}$ is zero. The aim of the feature reduction algorithm is to reduce the number of features being used in the image matching problem.

Our feature reduction algorithm takes the number of features to be eliminated as the input. The algorithm eliminates the features by using up to two methods. If the first method is sufficient to reduce the number of features to its target value, then the second method is not used. Only when the first method has been exhausted and further feature elimination is needed to meet the target value on the number of features, then the second method is used. We begin with the first method.

4.2.1 *Similarity Based Feature Reduction*

The first method uses a coarse-grained method to identify features that do not help us to differentiate between the images by calculating pairwise similarities between features. Consider the vector $\vec{z}_q = [z_{1,q}, z_{2,q}, \ldots, z_{m,q}]$ which indicates whether the q-th feature is present or absent in the m images. For example, if $m = 4$ and $\vec{z}_q = [1,1,0,0]$, then it means that the q-th feature is found in the first and second images, but not in the third and fourth images.

In order to identify features that are similar, we will work with a pair of features at a time. In order to fix the idea, we first present an example assuming that $m = 4$. We assume that for the q-th feature, we have $\vec{z}_q = [1,1,0,0]$. Now consider the r-th feature where $r \neq q$ and we have $\vec{z}_r = [1,1,0,0]$. In this example, we have $z_{i,q} = z_{i,r}$ for all i, it means both features q and r are either both present or both absent in each image. This implies that one of such features can be eliminated because one of them is redundant.

In general, consider two vectors $\vec{z}_q = [z_{1,q}, z_{2,q}, \ldots, z_{m,q}]$ and $\vec{z}_r = [z_{1,r}, z_{2,r}, \ldots, z_{m,r}]$ for, respectively, the q-the and r-th features. If these two vectors $\vec{z}_q$ and $\vec{z}_r$ are similar, in the sense that $z_{i,q} = z_{i,r}$ for most i, then these two features provide similar information and we can also drop one of them. We therefore define a similarity measure between the two features by:

$$Sim(\vec{z}_q, \vec{z}_r) = \frac{\sum_i I(z_{i,q} = z_{i,r})}{m} \quad (3)$$

where $I()$ is an indicator function which takes on the value of 1 if the expression inside the parentheses is evaluated to be true. We define a threshold δ_{Sim} and we only consider elimination if $Sim(\vec{z}_q, \vec{z}_r) \geq \delta_{Sim}$, i.e. the vectors are sufficiently similar. If elimination is to be carried out, we make a random choice between q and r, as they are similar and contribute almost the same information to image matching.

We use this similarity measure to iteratively to reduce the number of features. For all the features that are still retained in the dictionary, we compute the similarity measure $Sim(\vec{z}_q, \vec{z}_r)$ for every pair of features q and r. If there exist pairs of features whose similarity measure exceeds the threshold δ_{Sim}, we choose the pair of features which has the highest similarity measure (ties are broken randomly) and eliminate one of the features in the pair. We repeat this elimination process until: (1) If we reach the target number of features, the algorithm is terminated; or (2) If all pairs of remaining features have a similarity measure below δ_{Sim} and the target number of features has not been met, the algorithm proceeds to use the second method to eliminate features.

4.2.2 *Mutual Information Based Feature Reduction*

The second method is to reduce the number of features by using the concept of mutual information. We start by recalling a result from Information Theory on how the choice of features impacts on the classification error in supervised learning. Consider a classification problem with m classes denoted by $C = \{c_1, ..., c_m\}$ and n possible features indexed by $1, 2, ..., n$. Let $\mathcal{F} = \{1, 2, ..., n\}$ denote the set of all features. The number of features n can be large and we may only want to select a subset of features from $\mathcal{F}$ for classification. It is obvious that different subsets of $\mathcal{F}$ gives different classification error, but how we want to know which subset is better. Let F_1 and F_2 be two different subsets of $\mathcal{F}$, and we will call them feature sets. We use MI_j to denote the mutual information between the classes C and feature set F_j. A result from Information Theory says that if $MI_1 > MI_2$, then the use of feature set F_1 will lead to a smaller classification error, and vice versa [21]. This means that a good feature set should maximise the mutual information between the feature set and the classes. The importance of this result is that it gives us a method to evaluate the goodness of different feature sets by calculating mutual information. However, it is difficult to use this result in practice because we will need to have the joint probability distribution of all the features in a feature set, which is difficult to obtain empirically. We therefore use a method which requires only the joint probability distribution of a pair of features. In other words, we only consider a pair of features at a time.

The proposed method requires performing the same calculation over each pair of features. Let q and r be a pair of features. We want to calculate the mutual information between the set of classes C and the feature set $\{q, r\}$. The calculation of mutual information requires the probability distribution of the features. Let $Z_{i,q}$ denote a binary random variable which can take on values from $\{0, 1\}$. Let $p(Z_{i,q} = 1)$ (resp. $p(Z_{i,q} = 0)$) denote the probability that the q-th feature is present (absent) in image class c_i where an image class consists of all the images collected at a particular location. We use $p(Z_{i,q})$ as a short hand to denote that probability distribution $p(Z_{i,q} = z_{i,q})$. The mutual information between the set of all classes C and the feature set $\{q, r\}$, which uses both features q and r is:

$$MI(C;\{q,r\}) = \sum_i \sum_{z_{i,q},z_{i,r}} p(c_i, Z_{i,q}, Z_{i,r}) log(\frac{p(c_i, Z_{i,q}, Z_{i,r})}{p(Z_q, Z_r)p(c_i)}) \tag{4}$$

where $p(c_i, Z_{i,q}, Z_{i,r})$ is the joint probability distribution of features q, r and classes C; and $p(Z_q, Z_r)$ denotes the marginal probability over image index i.

Instead of using both features q and r, we consider the possibility of merging binary features using logical AND and NOT operations. Let $z_{i,qr}$ denote the merged feature which is also binary. There are four ways that we can define the merged feature $z_{i,qr}$:

- $z_{i,qr} = 1$ if $z_{i,q} = 1$ and $z_{i,r} = 1$; otherwise $z_{i,qr} = 0$
- $z_{i,qr} = 1$ if $z_{i,q} = 1$ and $z_{i,r} = 0$; otherwise $z_{i,qr} = 0$
- $z_{i,qr} = 1$ if $z_{i,q} = 0$ and $z_{i,r} = 1$; otherwise $z_{i,qr} = 0$
- $z_{i,qr} = 1$ if $z_{i,q} = 0$ and $z_{i,r} = 0$; otherwise $z_{i,qr} = 0$

The mutual information between the classes and the merged feature is given by

$$MI(C;qr) = \sum_i \sum_{z_{qr}} p(c_i, Z_{i,qr}) log(\frac{p(c_i, Z_{i,qr})}{p(Z_{qr})p(c_i)}) \tag{5}$$

The merging of features will lead to loss of information, so we expect $MI(C;\{q,r\}) \geq MI(C;qr)$. However, if the gap between $MI(C;\{q,r\})$ and $MI(C;qr)$ is small enough, we expect the merged feature still contains the information in the two individual features combined. For this merging, we define a positive threshold δ_{MI} and only proceed with merging if $MI(C;qr) \geq MI(C;\{q,r\}) - \delta_{MI}$, i.e. the use of the merged feature does not lead to significant decrease in mutual information.

The elimination of features using the this method proceeds iteratively as well. For each pair of features q and r that are still retained in the vocabulary and for each of the four ways to merge q and r, we compute the decrease in mutual information $MI(C;\{q,r\}) - MI(C;qr)$, respectively. If at least one decrease of mutual information is less than or equal to δ_{MI}, the algorithm chooses the pair of features and the way of merging that lead to the least decrease in mutual information, and the algorithm proceeds to merge those two features. The algorithm iterates until either no more features can be merged or the target number of features is reached. To evaluate this innovation, we will study the performance of this feature reduction algorithm in Section 5.2.

4.3 Step Detection and Avoiding Blurry Images

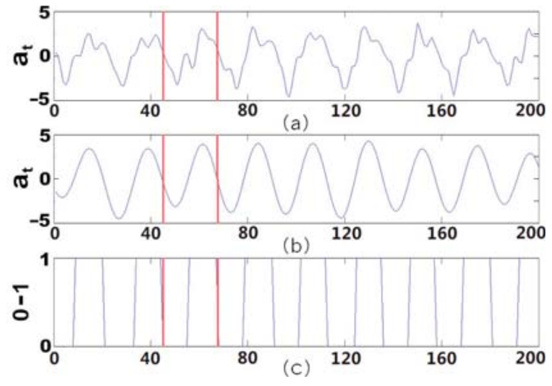


Figure 3: Step detection using time series data of acceleration (a) raw sensor readings (b) filtered sensor data (c) binarised data for step cycle detection

Our localisation method NaviGlass uses the readings from the accelerometer and gyroscope to determine, respectively, the step length and the heading. The algorithms we use are fairly standard. For step detection, let $a_{x,t}$ denote the acceleration at time t in the x-direction etc. We first compute the total acceleration $a_t = \sqrt{a_{xt}^2 + a_{yt}^2 + a_{zt}^2}$ at each sampling time t. Fig. 3(a) shows a trace of a_t from one of the experiments. We apply a low-pass filter with a cut-off of 6Hz to filter this trace to obtain a smoother trace shown in Fig. 3(b). We then apply a threshold to binarise this smoothed trace to obtain Fig. 3(c). A step cycle is a combination of a consecutive 0 and 1, shown between red lines in Fig. 3. Similarly, we can obtain the heading change from gyroscope when a step is detected by projecting the measurements to the world coordinate system for more accurate angle change [15].

In order to reduce the chance of blurry images being taken, the smart glasses will only take an image when the total acceleration is not too large. When the smart glasses want to take an image, they first determine the previous time instance at which the total acceleration peaked. They then compare the current time against the last time peak acceleration occurred, if the gap is less than a threshold 0.1s, then we delay the capture the image until 0.1s has passed from the previous peak. This threshold of 0.1s is not likely to cause much delay. Our observation is that 88% of a step lasts for 0.6 ± 0.2s and with the sampling interval for images set at 2s, an additional delay of 0.1s may extend the sampling interval to up to 2.1s, which is a negligible increase.

Another occasion that blurry images can be produced is when the subject makes a turn. This is because the camera takes some time to use its auto-focus mechanism after a turn is made. We therefore avoid taking an image soon after the subject has turned and we do this based on the gyroscope readings.

4.4 Particle filters and data fusion

In order to cope with noise in the IMU sensors, we use particle filters [1] to obtain a more stable estimate. Our particle filter uses the step length and heading as the input. The filter assumes that the noise in the IMU sensors are zero mean Gaussian. We assume the particle filter uses N particles. At each time t, each particle is characterised by a quadruple. For the i-th particle at time t, the particle is $P_{i,t} = (x_t^i, y_t^i, \theta_t^i, w_t^i)$ where (x_t^i, y_t^i) are the x-and-y coordinates of the particle, θ_t^i is the heading of the particle and w_t^i is the weight of the particle. Note that at each time t, the sum of weights is normalised, i.e. $\sum_i w_t^i = 1$. The localisation position estimate of the subject at time t is the weighted average of the positions of the N particles. The estimated x- and y-coordinates of the subject at time t are, respectively, $\sum_i w_t^i x_t^i$ and $\sum_i w_t^i y_t^i$.

Note that particle filters have been used together with floor plan of a target area to improve localisation estimate [26]. This can be done by setting the weight those particles that move in a unrealistic manner, e.g. walking through walls or glass panels, to zero. In this paper, we want to focus on studying the improvement that can be made by using image matching. We therefore do not incorporate floor plan in the particle filter. However, floor plans can be easily incorporated into our particle filter to further improve the localisation estimate.

4.4.1 Drift Correction

A major problem with PDR is that the position estimate drifts over time. Our aim is to use image matching to correct this drift. Assuming that an image is taken at time t and our matching algorithm is able to match this image to its correct location. We can correct the drift by setting the position estimate at time t to the matched location of image. However, image matching is not perfect and error does occur. Also, in order to cope with the computation constraint, we reduce the number of features used in matching, this has the effect of increasing the matching error. We therefore need a method to counteract the possibility of a matching error.

Our method does not use the image matching result of one image to trigger the process of correcting the position estimate from the particle filter. Instead, we only trigger the correction if the matching of three consecutive images produces consistent result. Let Pos_1, Pos_2 and Pos_3 denote the position estimates from three consecutive images. Since these images are collected over a gap of about 2s, the distance between consecutive estimates cannot be more than 2.2m given the normal walking speed. We therefore check whether the distance between Pos_1 and Pos_2, as well as that between Pos_2 and Pos_3, is consistent with this prior knowledge. If yes, then we accept the last estimate Pos_3 as a correct estimate and use it to correct the position estimate from the particle filter. If the position estimates Pos_1, Pos_2 and Pos_3 are not consistent, we do nothing.

We now explain how we fuse the position estimate from Pos_3 with the particles' positions. Let $\hat{x}_t$ and $\hat{y}_t$ denote the position estimate from the image Pos_3. What we want to do is to make the weights of those particles that are close to $(\hat{x}_t, \hat{y}_t)$ to have a higher weight. Consider the i-th particle, we compute the distance between this particle and $(\hat{x}_t, \hat{y}_t)$, which is given by $Dis_t^i = \sqrt{(x_t^i - \hat{x}_t)^2 + (y_t^i - \hat{y}_t)^2}$. We adjust the weight of the ith particle as:

$$w_t^i = e^{-\frac{Dis_t^i}{r}} \tag{6}$$

where r is a tunable parameter; the bigger r is, the less effectively such update works. Once the new weights are calculated, they are normalised so that their sum is unity.

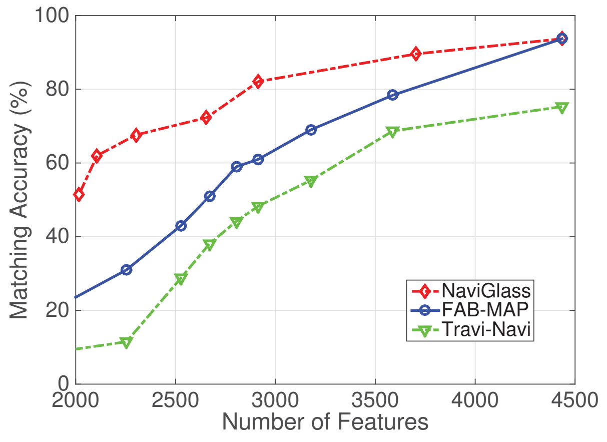
5 Evaluation

This section aims to evaluate the performance of our proposed localisation method NaviGlass. Our experiments were conducted on the Vuzix M100 smart glass platform [32] which runs the Android Operating System. These smart glasses is based on the Texas Instrument OMAP4460 processor running at 1.2GHz, 1 GB of RAM and 4GB of flash memory. Its sensors include a 3-axis accelerometer, a 3-axis gyroscope and a 5-MP camera.

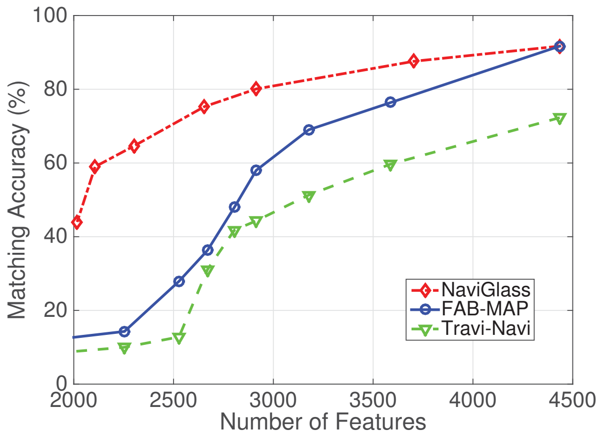
5.1 Experimental Overview

We implemented NaviGlass on the Vuzix M100 Glasses. We implemented our own PDR module which provides step counting and heading estimation, see Section 4.3. NaviGlass uses image matching to correct the PDR location estimates as described in Section 4. The image matching is based on the FAB-MAP framework. We based our work on the open-source FAB-MAP code in the OpenCV library (Version 2.4.9) [24]. The OpenCV code is written in C++ and we use the Java Native Interface (JNI) for interfacing. The FAB-MAP code offers a number of different choices of image descriptors and we have chosen to use SURF [3] image descriptor for efficiency and accuracy.

We compare NaviGlass against Travi-Navi, which is the state-of-the-art localisation method which uses both PDR and images. We implemented Travi-Navi on the Vuzix platform. The PDR module in our Travi-Navi implementation is identical to that of NaviGlass. The image matching module of TraviNavi uses the BoW framework [8] followed by a linear SVM classifier [11]. We used the SVM code from [4]. For fair comparison, both Travi-Navi and NaviGlass use the



(a) Results of testing data from same day



(b) Results of testing data from different days

Figure 4: Matching accuracy vs. number of features

same SURF image descriptor. Note that Travi-Navi also uses WiFi fingerprint and magnetic distortions to correct the position estimates, however we have not incorporated them into our implementation. This is because WiFi coverage is weak in some application scenarios such as underground parkings and stadiums, and WiFi fingerprints and magnetic field are known to be dynamic over time and therefore difficult to collect and maintain. Furthermore, NaviGlass can also benefit from WiFi fingerprint and magnetic field distortion correction similar to Trivi-Navi if they are available.

We conducted the experiments on our university campus. Ten subjects (2 females and 8 males) participated in data collection and system validation. Further experimental details will be provided later.

5.2 Image Matching Accuracy

This section evaluates the accuracy of image matching. The experiments were conducted along a 125m long looped path within a laboratory. The subjects were asked to walk along this path wearing the Vuzix smart glasses, which were programmed to take 10 images per second with a resolution of 320×240. The training images were obtained by one subject walking along the designated path. We collected two sets of test images. The first set of test images was collected on the same day that training images were collected, while the second test image set was collected a week after training occurred. In both sets of test images, all 10 subjects walked along the designated path for image collection. The ratio of the number of images in a test set to the number of images in the training set is approximately $10 : 1$.

Our aim is to compare the image matching accuracy of NaviGlass, FAB-MAP and Travi-Navi with varying number of features. We first explain how we vary the number of features. For FAB-MAP, the number of features is controlled by using a threshold in the Hessian detector of the SURF features [3]. For NaviGlass, we start with a large number of SURF features (about 4500) and use the feature reduction algorithm in Section 4.2 to reduce the number of features. Travi-Navi uses the same features as FAB-MAP. Note that NaviGlass uses maximum a posteriori probability to determine a match while Travi-Navi uses SVM.

In this section, we say a test image is a *correct match* if the determined location of the test image is within 3 steps (approximately 1m) from its true location. We report the accuracy of each method by using the percentage of correct matches, which is the total number of correct matches divided by the total number of images in a test image set.

Figures 4(a) and 4(b) show the matching accuracy for the test sets which were collected, respectively, on the same day and one week later. Unsurprisingly, the matching accuracy increases with larger number of features. For both test sets, NaviGlass has the best performance, followed by FAB-MAP and Travi-Navi. For example, when about 2000 features were used, the accuracy of NaviGlass is 30% higher than that of Travi-Navi. The use of test image set which was collected one week after training data collection enables us to test the temporal robustness of image matching. The matching accuracy of three methods for the one week old test set is lower than the same-day test set. The matching accuracy of NaviGlass is 5% lower, while FAB-MAP and Travi-Navi are respectively, 11% and 10% lower.

5.3 The Impact of Image Resolution

We investigate the impact of image resolutions on matching accuracy and computation time on the Vuzix M100 platform. Figure 5 shows the matching performance of NaviGlass and Travi-Navi when the number of features is 2107. We use 6 different image resolutions, from 960×720 down to 80×60. As expected, the matching performance of both methods decreases when the image resolution decreases. The decrease in accuracies from 960×720 resolution to 160×120 resolution appears to be steady; the drop in accuracy is pretty drastic after a resolution of 160×120. For NaviGlass, the accuracies of matching 960×720 and 160×120 images are, respectively, 74% and 53%. The image matching accuracies of Travi-Navi is about 7-10% lower than that of NaviGlass.

Table 1 shows the time needed to process one image for

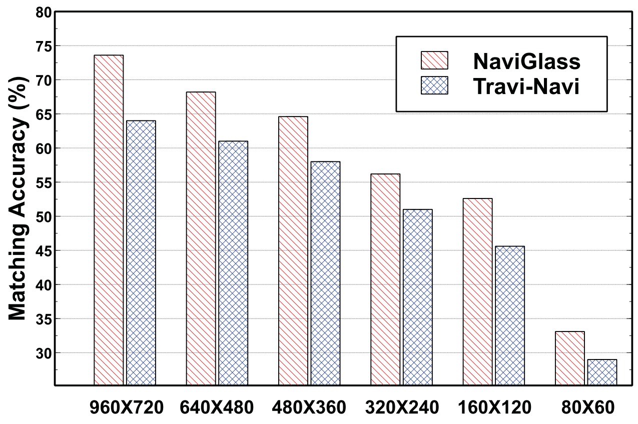


Figure 5: Matching accuracy vs. image resolution

NaviGlass, FAB-MAP and Travi-Navi for different image resolutions. The processing time includes the time to calculate the SURF features and the time to perform an image matching. The results in the table is based on using 2017 features. Note that all three methods require roughly the same processing time. For NaviGlass and a resolution of 160×120, we require 1.73s of processing to achieve an accuracy of 53.3%, which is 8% higher than TraviNavi for the same image resolution. If we increase the resolution to 320×240, the computation time increases to 5.8s giving an accuracy of 56.7%. This shows that a 3.4-fold increase in computation time earns us little increase in accuracy. For our further evaluation, we have chosen to use a resolution of 160×120.

Table 1: Processing time per image.

Resolution	NaviGlass	FAB-MAP	Travi-Navi
960×720	28980ms	28477ms	30844ms
640×480	15578ms	16787ms	16023ms
480×360	8818ms	9845ms	9144ms
320×240	5801ms	6155ms	6451ms
160×120	1730ms	1865ms	1802ms
80×60	1301ms	1221ms	1355ms

5.4 The Impact of Camera Sampling Rates

This section evaluates the effect of image sampling rates on localisation accuracy. For this evaluation, we marked the path in the target area and asked the subjects to walk along the marked path. We performed the experiments in two target areas: in a laboratory and in a library. We want to test 9 different sampling intervals: 0.5, 1, 1.5, 2, 4, 6, 8, 10 and 12 seconds. For the experiments in the laboratory, the 10 subjects walked the path 6 times and each time a different sampling interval was used; the sampling intervals used were 2, 4, 6, 8, 10 and 12 seconds. For the experiments in the library, the 10 subject walked along the path only once using a sampling interval of 0.5s purely for collecting the images because the length of the path in the library is fairly long.

For the experiments in the laboratory with sampling intervals in 2–12 seconds, the localisation was performed on the smart glasses. Since the smart glasses do not have enough computation resources to process the images if sampling intervals in 0.5-1.5s are used, the localisation was performed off-line. For the experiments in the library, all the localisation computation were conducted off-line by sub-sampling the images.

For localisation methods, we use NaviGlass and FAB-MAP. Note that FAB-MAP is a purely vision based localisation method and PDR is not used in FAB-MAP. The goal of this evaluation is to compare a PDR + vision localisation algorithm against a pure-vision based localisation algorithm.

Fig. 7 shows the average localisation error for NaviGlass and FAB-MAP for all the 9 different sampling intervals. The results are obtained by averaging the results in the laboratory and in the library. Note that the average results in the laboratory and the average results in the library are similar to those in Fig. 7. The figure shows that NaviGlass can achieve a much lower localisation accuracy for a given sampling interval. It also shows that if one can optimise the computation so that sampling interval can be reduced to 0.5s, then one can realise a sub-metre accuracy.

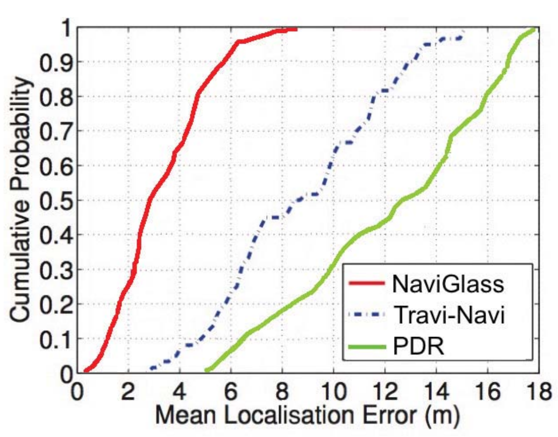
Our system outperforms pure vision-based method FAB-MAP for all sampling intervals. Note that FAB-MAP was originally designed for localisation of robots in an outdoor environment. There are two key differences between outdoor robot localisation and indoor smart glass localisation. First, robots can use multi or panoramic cameras to capture images but such facilities are not present for smart glass. Second, the outdoor environment, especially the dense urban environment, offers many unique visual features to identify a location. The same cannot be said for indoor environment, especially the library environment in which we tested our localisation environment, where the scenes can be very similar. This makes the library environment challenging for localisation with smart glasses.

5.5 Comparison with Travi-Navi

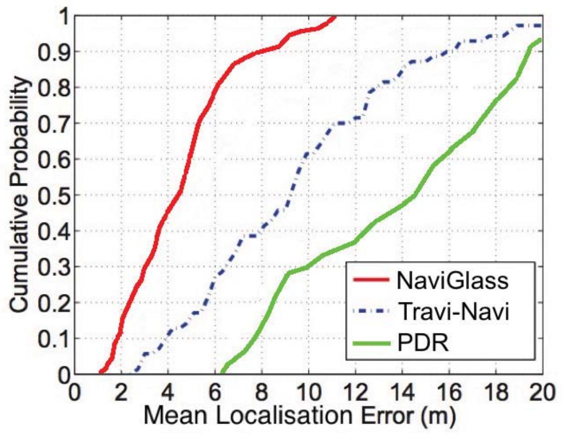
We compare the performance of NaviGlass against Travi-Navi using the same test data. Note that we used localisation accuracy (the concern of this paper) instead of pathway accuracy as in Travi-Navi.

5.5.1 Accuracy

We conducted experiments in the laboratory and in the university library, covering $948m^2$ and $3926m^2$, respectively. Ten subjects were asked to walk along two specific routes, shown in Fig. 8 and Fig. 9, respectively. During the experiments, the smart glasses collected images of resolution 160×120 at 0.5Hz. Figure 6(a) shows the cumulative probability distribution of localisation error for NaviGlass, Travi-Navi and PDR. PDR gives the worst accuracy because of accumulation of error. With the help of image matching, both NaviGlass and Travi-Navi performed better than PDR. However, our proposed NaviGlass outperforms Travi-Navi because of better image matching accuracy. NaviGlass achieves mean localisation error of 3.3m, which is 64% less than that of Travi-Navi. Fig. 6(b) shows the localisation error distribution for the library. It can be seen that the performance in the library is worse than that in the laboratory.



(a) Results in the lab



(b) Results in the university library

Figure 6: Performance of our system and Travi-Navi

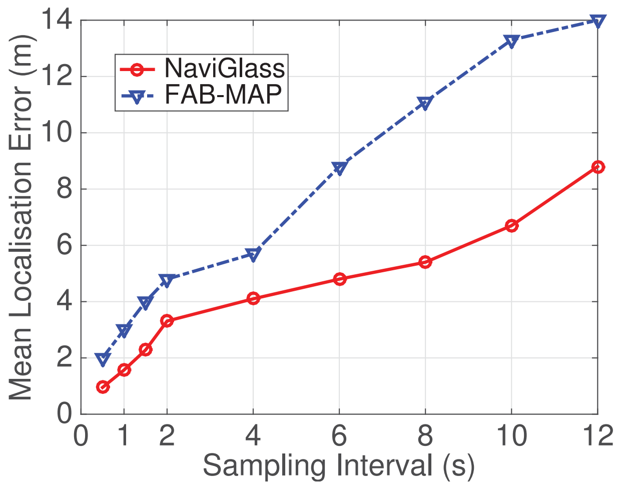


Figure 7: Performance of different sampling rates

The library environment is more challenging for a few reasons: (1) The path is much longer and has more turns. (2) The scenes are very similar at some places, e.g. in an aisle with bookshelves on both sides. (3) The scenes are more dynamics with people and books changing. Despite these challenges, NaviGlass again outperforms Travi-Navi.

Figures 8 and 9 compare the paths obtained from PDR and NaviGlass in the laboratory and the library. The drifting of the localisation estimates for PDR is clearly visible. These figures also show that NaviGlass is capable of correcting the drifts.

5.5.2 *Resource Consumption*

Table 2 compares the resource consumption of NaviGlass and Travi-Navi. The figures in the table are obtained from the average of 20 walks by the 10 subjects.

The first section of Table 2 shows the time needed to match an image and the processing time of an image. Since the time to process an image consists of two major tasks: feature extraction and image matching, we see that feature extraction has now become the dominant computation component because we have optimised the time for image matching using feature reduction.

Table 2: Resource consumption on Vuzix Smart Glasses

	NaviGlass	Travi-Navi
Matching Time	115ms	87ms
Total Time	1730ms	1802ms
Display Energy	268mJ	259mJ
Camera Energy	175mJ	170mJ
IMU Energy	501mJ	532mJ
CPU Energy	1642mJ	1708mJ
Expected Battery Life	0.87h	0.78h

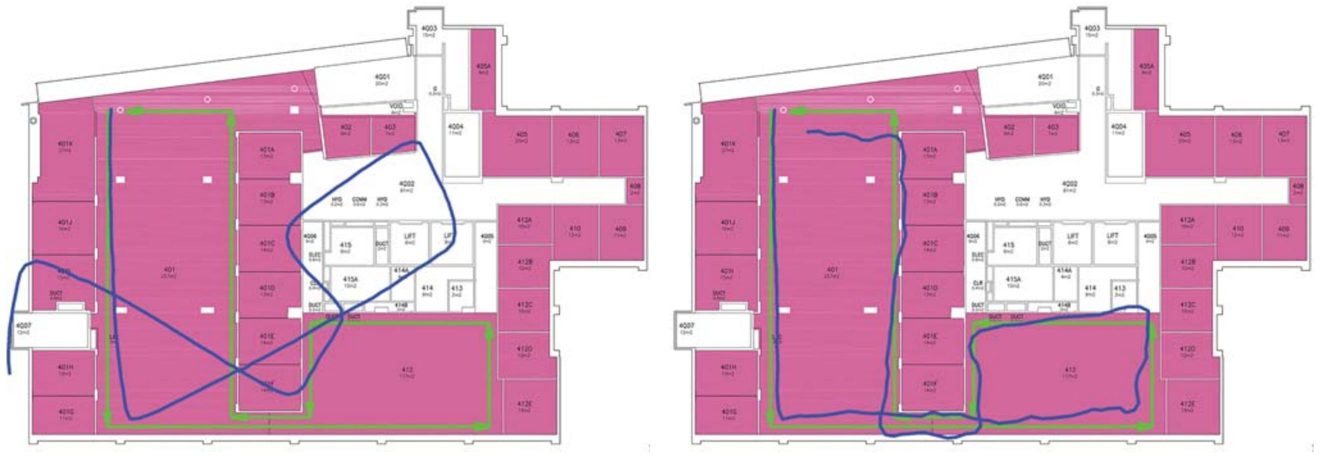
The middle section of Table 2 shows the average energy consumption for display, IMU, camera and CPU. The energy consumption is obtained by using PowerTutor [39] which is an Android application that reports power consumption in real-time. It can be seen that the energy consumption is dominated by that of CPU.

Vuzix M100 comes with a battery with a nominal capacity of 550mAh. By using the average current (e.g. 632.2mA for sampling rate 0.5Hz for NaviGlass), we can compute the expected battery life for our localisation applications in the last section of Table 2. In practice, the battery life is shorter due to battery ageing.

Overall, Table 2 shows that the computation time, energy consumption and expected battery life of both NaviGlass and Travi-Navi are similar. Since NaviGlass has a better localisation accuracy compared to Travi-Navi, our proposed localisation method is therefore superior.

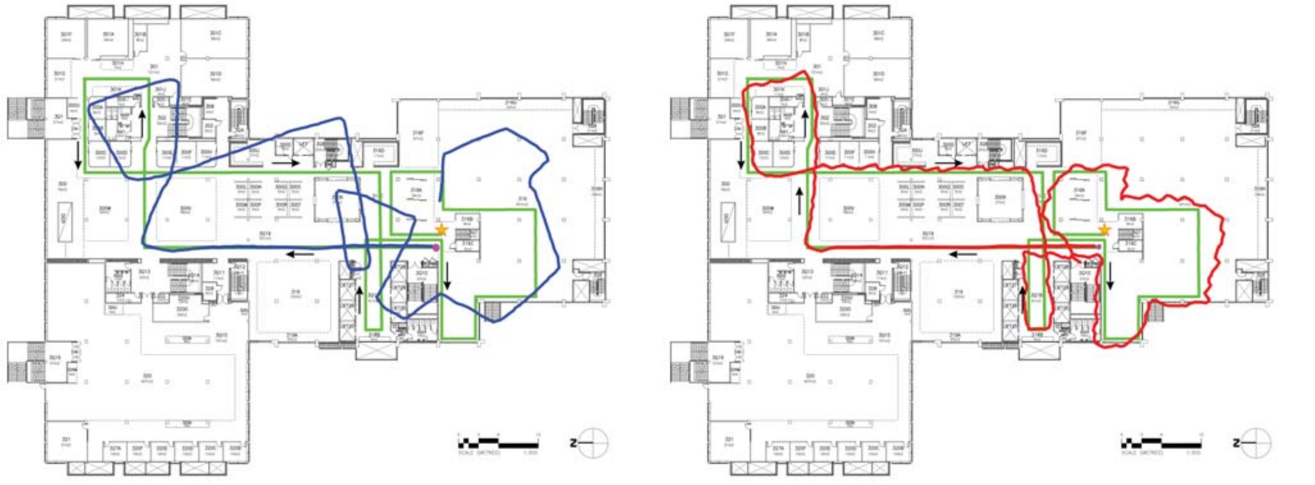
6 Conclusion

This paper proposes an indoor localisation method called NaviGlass for smart glasses. The key idea of NaviGlass is to



(a) Results of PDR (b) Results of our system

Figure 8: Experiments in lab, showing ground truth (green), results from PDR and NaviGlass (Best view in colour)



(a) Results of PDR (b) Results of our system

Figure 9: Experiments in the library, showing ground truth (green), results from PDR and NaviGlass (Best view in colour)

use PDR as the primary localisation method and to use image matching to correct the drift in PDR estimate. The key challenge in using image matching on smart glasses is the long computation time needed to match the images due to large feature space and limited computational resources on the smart glasses. We address this by proposing a feature reduction method which provides informative features for matching. We evaluate the performance of various component of NaviGlass, including image matching accuracy and computation time. We compare the performance of NaviGlass against Travi-Navi and find that NaviGlass outperforms Travi-Navi by a good margin.

7 References

[1] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 50(2):174–188, 2002.

[2] L. Atzori, T. Dessi, and V. Popescu. Indoor navigation system using image and sensor data processing on a smartphone. In *Optimization of Electrical and Electronic Equipment (OPTIM), 2012 13th International Conference on*, pages 1158–1163. IEEE, 2012.

[3] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.

[4] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[5] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan. Indoor localization without the pain. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, pages 173–184. ACM, 2010.

[6] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, 1968.

[7] J. Chung, M. Donahoe, C. Schmandt, I.-J. Kim, P. Razavai, and M. Wiseman. Indoor location sensing using geo-magnetism. In *Pro-*

ceedings of the 9th international conference on Mobile systems, applications, and services, pages 141–154. ACM, 2011.

[8] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[9] M. Cummins and P. Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

[10] M. Cummins and P. Newman. Appearance-only slam at large scale with fab-map 2.0. *The International Journal of Robotics Research*, 30(9):1100–1123, 2011.

[11] R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using second order information for training support vector machines. *The Journal of Machine Learning Research*, 6:1889–1918, 2005.

[12] F. Fleuret. Fast binary feature selection with conditional mutual information. *The Journal of Machine Learning Research*, 5:1531–1555, 2004.

[13] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan. Towards wearable cognitive assistance. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*, pages 68–81. ACM, 2014.

[14] J. Haverinen and A. Kemppainen. Global indoor self-localization based on the ambient magnetic field. *Robotics and Autonomous Systems*, 57(10):1028–1035, 2009.

[15] F. Hong, Y. Zhang, Z. Zhang, M. Wei, Y. Feng, and Z. Guo. Wap: Indoor localization and tracking using wifi-assisted particle filter. In *Local Computer Networks (LCN), 2014 IEEE 39th Conference on*, pages 210–217. IEEE, 2014.

[16] Y. Jin, H.-S. Toh, W.-S. Soh, and W.-C. Wong. A robust dead-reckoning pedestrian tracking system with low cost sensors. In *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*, pages 222–230. IEEE, 2011.

[17] Y. Kameda and Y. Ohta. Image retrieval of first-person vision for pedestrian navigation in urban area. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 364–367. IEEE, 2010.

[18] H. Kawaji, K. Hatada, T. Yamasaki, and K. Aizawa. An image-based indoor positioning for digital museum applications. In *Virtual Systems and Multimedia (VSMM), 2010 16th International Conference on*, pages 105–111. IEEE, 2010.

[19] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. A reliable and accurate indoor localization method using phone inertial sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 421–430. ACM, 2012.

[20] H. Lim, L.-C. Kung, J. C. Hou, and H. Luo. Zero-configuration, robust indoor localization: Theory and experimentation. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12. IEEE, 2006.

[21] J. Lin. Divergence measures based on the Shannon entropy. *Information Theory, IEEE Transactions on*, 37(1):145–151, 1991.

[22] Y. Lu, I. Cohen, X. S. Zhou, and Q. Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th international conference on Multimedia*, pages 301–304. ACM, 2007.

[23] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. Landmarc: indoor location sensing using active rfid. *Wireless networks*, 10(6):701–710, 2004.

[24] OpenCV. `http://opencv.org/`.

[25] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1226–1238, 2005.

[26] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: zero-effort crowdsourcing for indoor localization. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 293–304. ACM, 2012.

[27] S. Rallapalli, A. Ganesan, K. Chintalapudi, V. N. Padmanabhan, and L. Qiu. Enabling physical analytics in retail stores using smart glasses. In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 115–126. ACM, 2014.

[28] J. Rivera-Rubio, I. Alexiou, and A. A. Bharath. Appearance-based indoor localization: A comparison of patch descriptor performance. *Pattern Recognition Letters*, 2015.

[29] F. Seco, C. Prieto, J. Guevara, et al. A comparison of pedestrian dead-reckoning algorithms using a low-cost mems imu. In *Intelligent Signal Processing, 2009. WISP 2009. IEEE International Symposium on*, pages 37–42. IEEE, 2009.

[30] M. Susi, V. Renaudin, and G. Lachapelle. Motion mode recognition and step detection algorithms for mobile phone users. *Sensors*, 13(2):1539–1562, 2013.

[31] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 2, pages 1023–1029. Ieee, 2000.

[32] Vuzix. M100 Smart Glasses - Enterprise. `http://www.vuzix.com/consumer/products_m100/`.

[33] H. Wang, X. Bao, R. Roy Choudhury, and S. Nelakuditi. Visually fingerprinting humans without face recognition. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pages 345–358. ACM, 2015.

[34] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: unsupervised indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 197–210. ACM, 2012.

[35] J. Xiong and K. Jamieson. Arraytrack: A fine-grained indoor location system. In *NSDI*, pages 71–84, 2013.

[36] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 197–206. ACM, 2007.

[37] M. Youssef and A. Agrawala. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218. ACM, 2005.

[38] L. Zhang, X.-Y. Li, W. Huang, K. Liu, S. Zong, X. Jian, P. Feng, T. Jung, and Y. Liu. It starts with igaze: Visual attention driven networking with smart glasses. In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 91–102. ACM, 2014.

[39] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proceedings of the 8th IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pages 105–114. ACM, 2010.

[40] W. Zhang and J. Kosecka. Image based localization in urban environments. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 33–40. IEEE, 2006.

[41] Y. Zheng, G. Shen, L. Li, C. Zhao, M. Li, and F. Zhao. Travi-navi: Self-deployable indoor navigation system. In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 471–482. ACM, 2014.