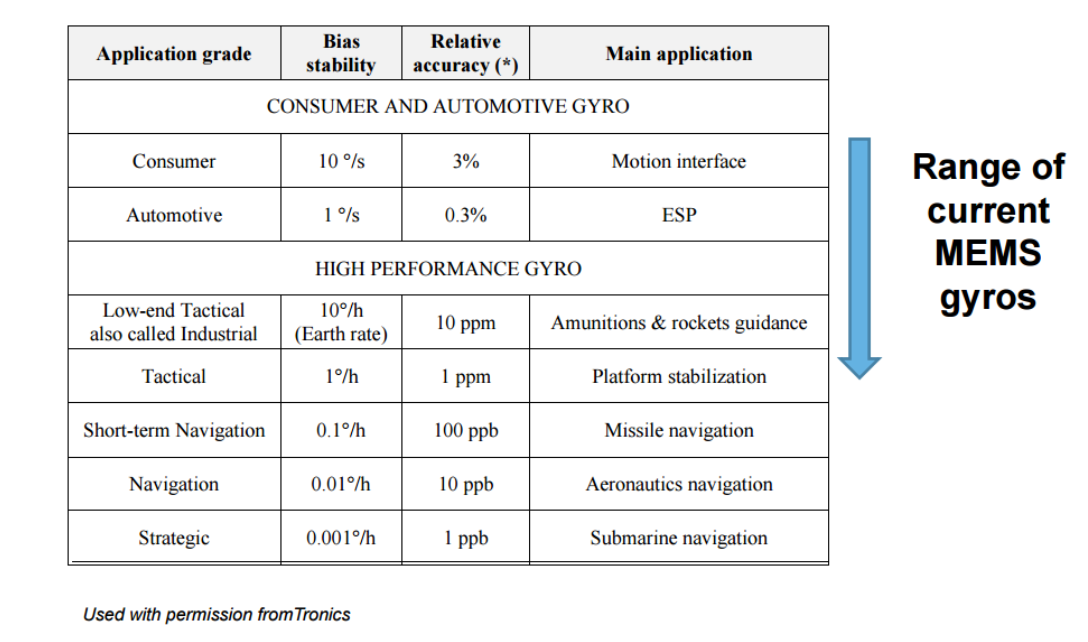
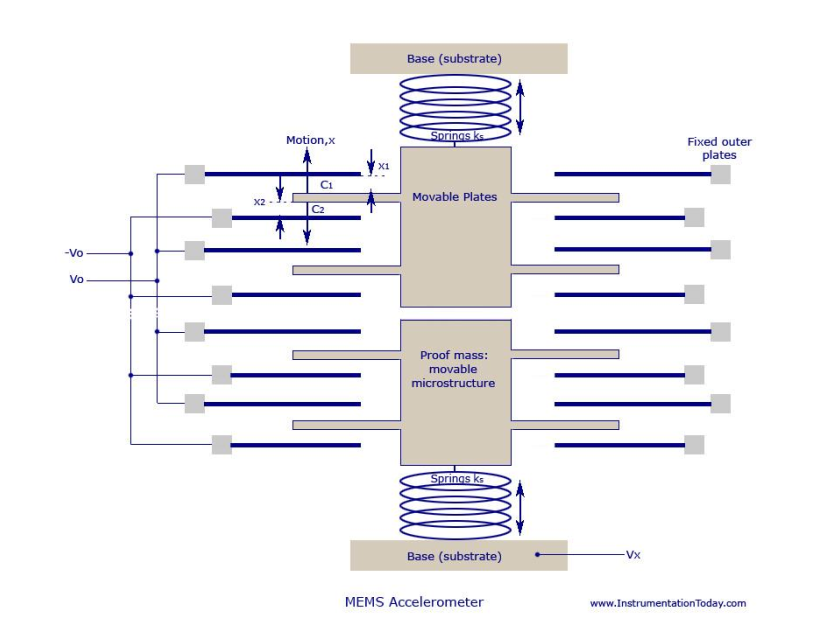
1. MEMS IMUInertial motion sensors typically contain three orthogonal accelerometers and three orthogonal gyroscopes, and sometimes three orthogonal magnetometer, measuring angular velocity, acceleration and magnetic field respectively. By processing signals from these devices it is possible to track the position and orientation of a device. Inertial motion sensors are the fundamental tools of pedestrian tracking. Their precision and reliability directly affects the quality of output

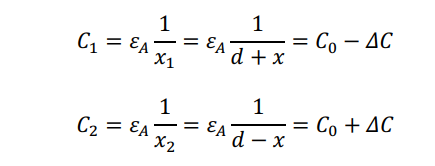


In the development of the Hubble Scope by NASA, a set of gas-bearing gyroscopes were used. It was claimed as the most accurate gyroscope in the world by NASA. But while better performance comes with more sophisticated systems, each gasbearing gyro is as big as a suitcase, not factoring in the external power source. In aviation and naval navigation, especially submarine navigation, a comparatively very precise and expensive ring laser gyroscope with better than 0.001 deg/hour bias uncertainty is used to capture the orientation data, which gives considerably reliable result. But like the gas-bearing gyroscope, a RLG (Ring Laser Gyroscope) has a diameter of over 50 cm. In the case of tracking pedestrians in a hospital, a low-cost sensor with limited size and power supply is required. Thus MEMS IMU is selected as the best suited sensor module for our purposes.

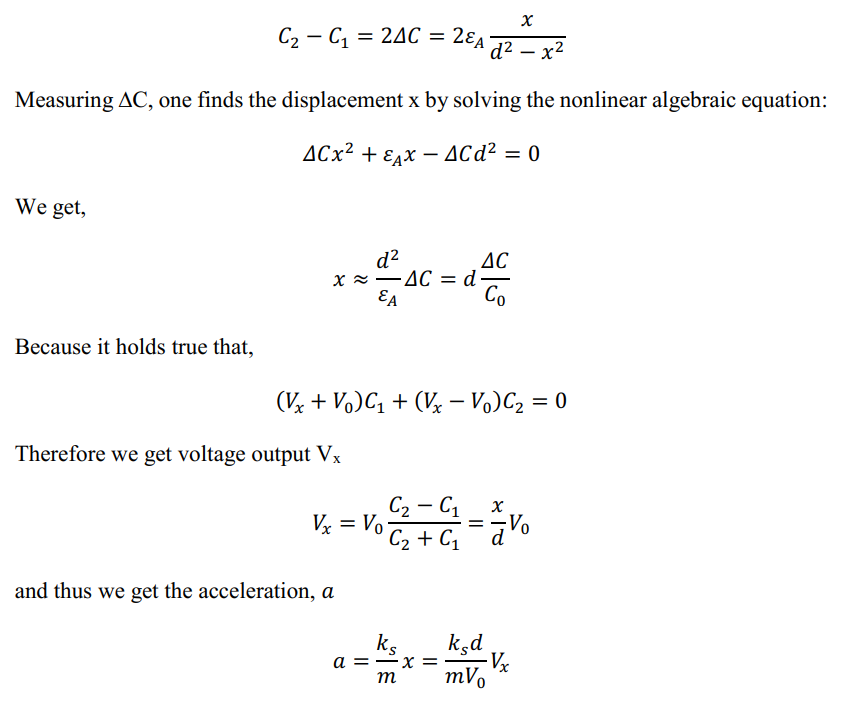
* 1. What is MEMS IMU?
  2. Drifts and Errors of MEMS IMU

MEMS devices are extremely microscopic devices and are vulnerable to many factors. Their performance is generally considered poor in accuracy for certain applications because of drifting, which is the nature of inertial sensors and which is more severe in MEMS. In order to understand the errors and drift, an understanding of the design of MEMS IMU is necessary.

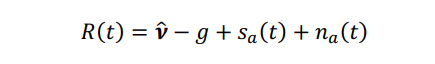


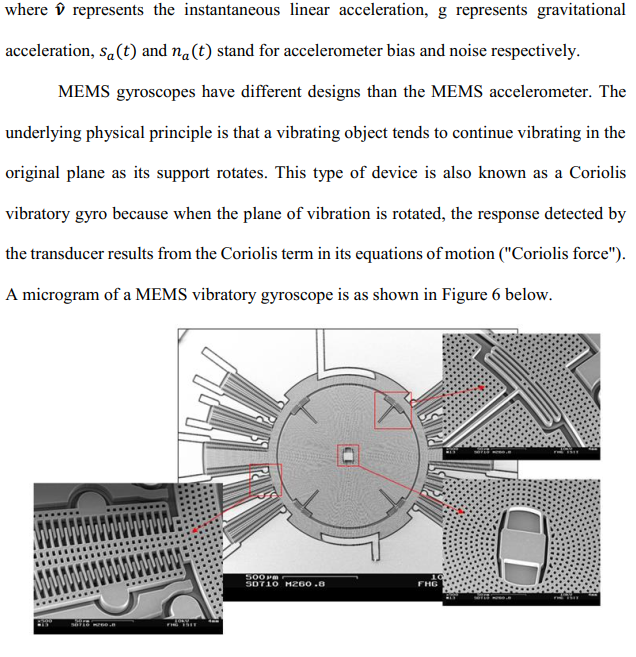
The MEMS accelerometer can be considered as a mass-spring system. Figure 5  
shows an illustration of a typical MEMS accelerometer on one axis. It is composed of  
movable proof mass with plates that is attached through a mechanical suspension system  
to a reference frame. When the object has an acceleration in the direction to which the  
spring is fixed, the plate moves and the spacing between the comb structure and adjacent  
outer fixed plates changes. By measuring the capacitance difference on each side of the  
plate, the deflection of the proof mass is calculated [4].  


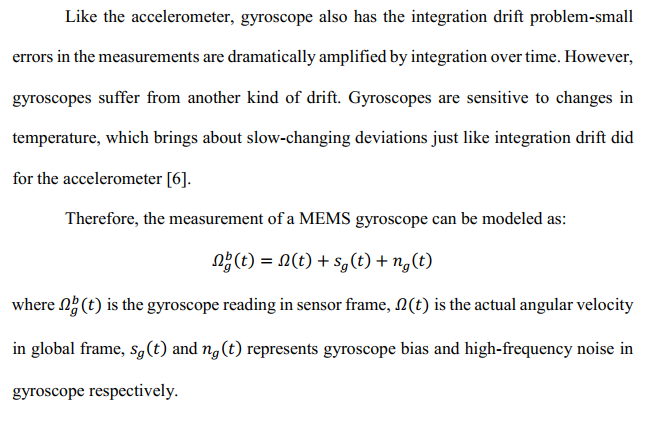
So,



We notice that in fact the accelerometer does not measure linear acceleration. It picks up forces applied on the proof mass and measure the displacement of the mass to calculate acceleration. The force of gravity is always picked up by the accelerometer and is taken as acceleration. An accurate attitude of the sensor has to be estimated in order to subtract the gravity factor on all three axes so as to obtain an estimate of linear acceleration. Otherwise small errors in the measurement of acceleration are integrated into progressively larger errors in velocity, which are compounded into still greater errors in position. Since the new position is calculated from the previous calculated position and the measured acceleration and angular velocity, these errors accumulate roughly proportionally to the time since the initial position was input [5]. This is known as the notorious integration drift problem of inertial navigation system, which is greatly reduced using the approach proposed in this thesis.  
Therefore, the measurement of an accelerometer can be modeled as:



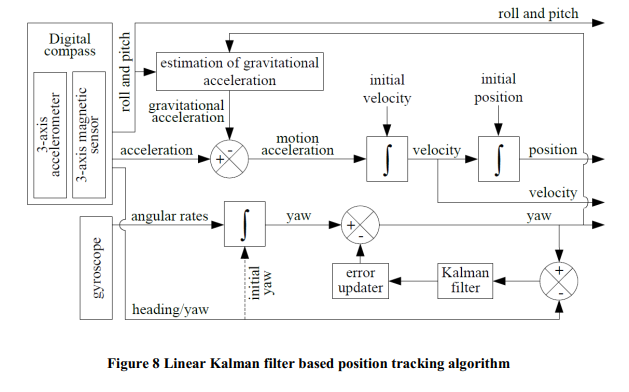


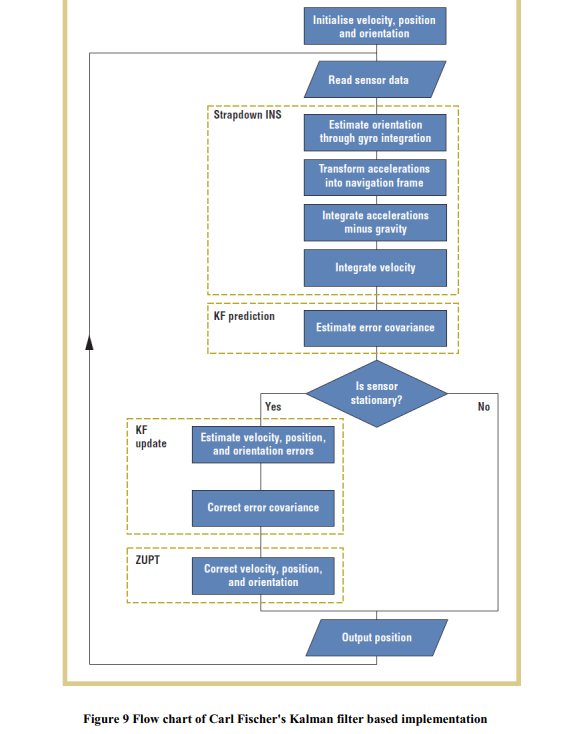


1. AHRS algorithmMEMS IMUs are vulnerable to various kinds of noise and errors. Accelerometers  
   are extremely sensitive to attitude changing and impact forces while gyroscopes are  
   sensitive to temperature changes and suffer from a slow-changing bias. To summarize, accelerometers have poor dynamic features and gyroscopes have poor static features.  
   Therefore an AHRS (Attitude and Heading Reference System) algorithm is needed to  
   fuse the data from different sensors to overcome the drawbacks of each of them and take the most reliable part from them respectively to give a best prediction of the actual status of the sensor. AHRS algorithm is the foundation of position estimation for the reason that gravity must be removed completely from the accelerometer to get linear acceleration. Only then can the integration be done without being concerned about the drift.  
   There are two main categories of AHRS algorithms. One is based on the Kalman  
   filter and the other is based on the so-called complementary filter.
   1. Kalman Filter

<----------(What is the Kalman Filter)----------->

A number of AHRS algorithms are developed based on the Kalman filter. Pedro  
Neto et al. proposed a position estimation algorithm based on the Kalman filter to correct the yaw [7]. Figure 8 shows the block diagram of that implementation.

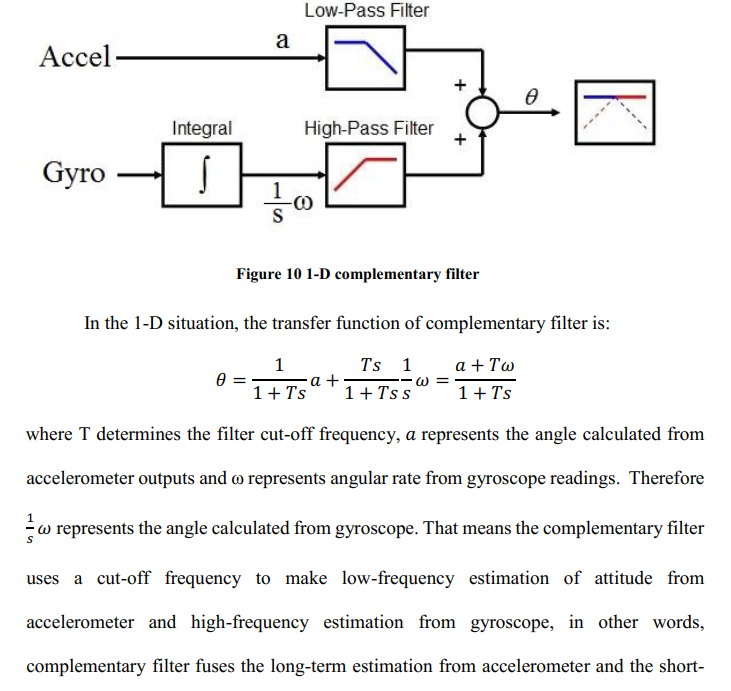


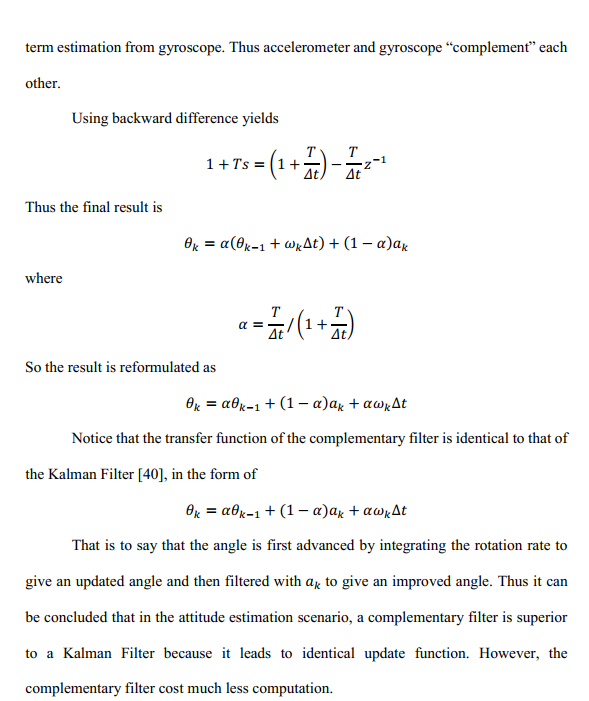


* 1. **Complementary Filters**

Since Kalman filter’s time complexity is too heavy a burden for mobile devices,  
algorithms with much lower computational complexity and almost equally efficiency are  
getting more acceptance, such as the complementary filters proposed by Robert Mahony  
[12].  
As is discussed above, accelerometers are sensitive to attitude changing and  
impact forces while gyroscopes are sensitive to temperature changes and suffer from

slow-changing bias. That means accelerometers have poor dynamic feature while gyros  
have poor static feature. The complementary filter takes this fact and make good use of  
each part. It fuses the accelerometer data and gyroscope data by first passing the  
accelerometer data through a 1st-order low pass filter, and then the gyroscope data  
through a 1st-order high pass filter, as is shown in Figure 10. Thus the complementary  
filter extracts the most reliable part of each sensor. Then a weighted average is taken  
from these two measurements.





1. Rotations

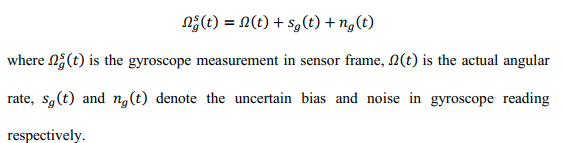
* What is the Rotations
* Application of Rotations.

1. Quaternions

* What is the Quaternions
* Application of Quaternions.

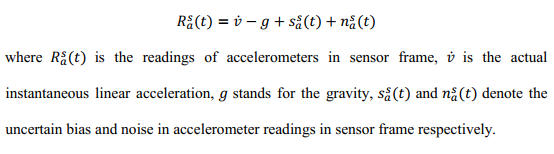
1. Sensor Noise Modelling
   1. Gyroscope Noise Model

Gyroscope measures the angular rate. However the main concern is the bias issue.  
It measures the rotation rate along with noise and bias about three orthogonally installed axis. For MEMS gyroscopes, temperature, impact force and other factors can cause uncertain bias which is hard to model. For simplicity, the gyroscope measurement can be modelled as:



* 1. Accelerometer Noise Model

Theoretically, accelerometer measures instantaneous acceleration only.  
However, practically, as described in section 1.2.2 Drifts and Errors of MEMS IMU, a MEMS accelerometer picks up all the forces applied on the proof mass. The acceleration readings are calculated from the mass of the proof mass and force applied on it.  
Therefore, gravitational acceleration with some added bias and noise are mixed in the readings. So the accelerometer measurement can be modelled as:



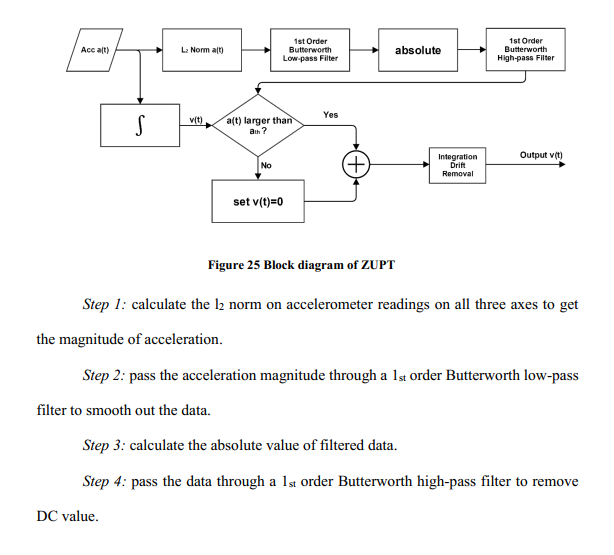
1. ZERO VELOCITY UPDATE
   1. Pedestrian Dead Reckoning

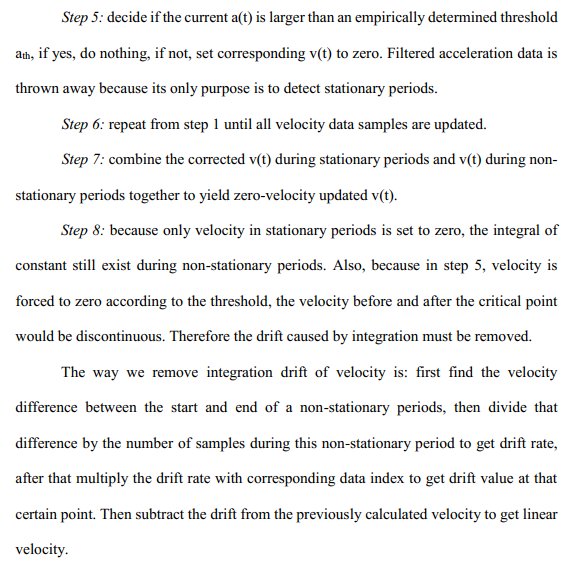
In navigation, pedestrian dead reckoning (PDR) is the process of estimating a  
pedestrian’s trajectory and position by using a previously determined or fixed position and advancing that position based upon known or estimated speeds or accelerations over time.  
Such a navigation system that tracks the location of a person is useful for finding and rescuing firefighters or other emergency first responders. As for example in our case, the caregivers can know the location of the patient that needs help immediately. It can also be employed in Location Based Services (LBS), mobile 3D audio and virtual reality applications.  
In the previous chapters, we saw that it was possible to obtain a satisfactory  
estimation of attitude using explicit complementary filters. An associated goal of this project is to obtain an estimation of location.

* 1. Zero Velocity Update (ZUPT)

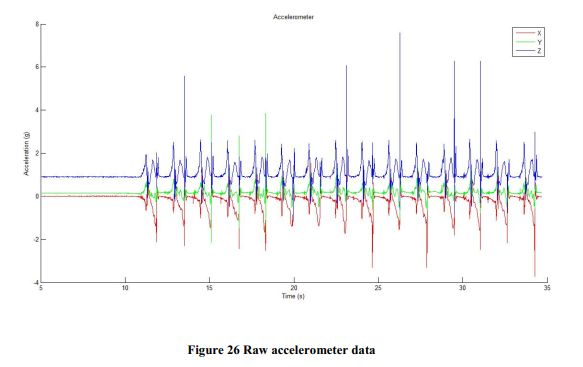
Conventional personal dead reckoning systems detect steps using a pedometer or an accelerometer. Then it takes magnetometer data to detect which direction is the pedestrian facing. Whenever a step is detected, it moves the position estimation forward by one step length to that direction. The step length is usually calculated by a waist- or limb-mounted sensor that detect angle of limb swing [28]. Some systems even need input of a pedestrian’s height to presume his step length. For these systems, step length is an average of previous detected steps. Therefore it suffers from lags and does not give satisfactory instantaneous position estimation. What is more, these systems always move  
the position estimate one step forward to the front; if the pedestrian steps sideways, the estimated position still goes to the front.  
In 1996, a DARPA (Defense Advanced Research Projects Agency) project  
proposed a method to detect steps using shoe-mounted sensors called zero-velocity update [29] (ZUPT), but the result was never published. The DARPA website denies public access. The first published research involving zero-velocity update was proposed by John Elwell [31]. Zero-velocity detection is a vital part in inertial navigation system.  
Because inertial sensors are subject to drift, if periods of stationarity is not detected from time to time, errors in acceleration would be integrated in to velocity and position that leads to drastic drift. Zero-velocity provides the required information to reset velocity [30].  
There are basically two approaches to detect zero velocity. One is using the  
knowledge of motion patterns of human to detect stance phase. Typically, such methods model walking as a repeating sequence of heel strike, stance, push off, and swing. These methods only work for walking and cannot detect movements like running, crawling or walking backward [30].  
The second and more generic option utilizes data from inertial sensor alone. It  
assumes that when the sensor is stationary (reading is 0), velocity is zero and angular rate is 0 as well. One concern is that if the accelerometer is moving at a constant speed, the algorithm would misjudge the motion as stationary. But based on our practice, since the accelerometer is very sensitive to accelerations, or, forces, and also because walking is a rather complex course of acceleration and deceleration, the detection of zero velocity never failed because of misjudgment. There are times that motion detection threshold  
was not set properly which led to misjudging. However this is a problem that can be easily avoided.

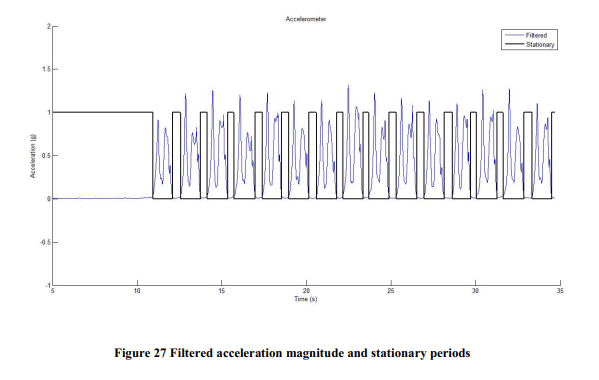
* 1. Implementation





* 1. Results





In Figure 27 the filtered accelerometer data as well as the detected stationary  
periods are shown.

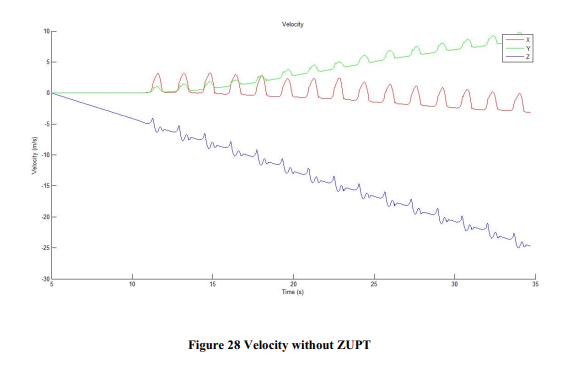


Figure 28 shows the velocity data without zero-velocity correction. Steps are still recognizable but we can vividly see the drift caused by 1st order integration of constant value

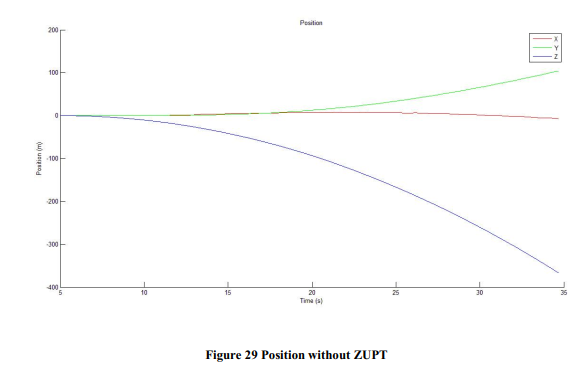


Figure 29 shows the position result without zero-velocity correction. The position estimation drifts so drastically that we cannot even find ripples of steps in the curve.

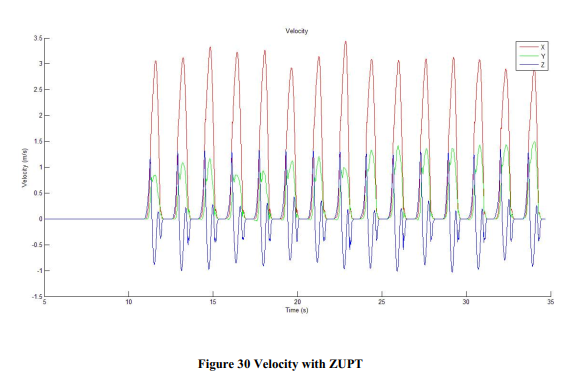


Figure 30 shows the velocity estimation with ZUPT. We can see the graph is  
reasonably satisfactory.

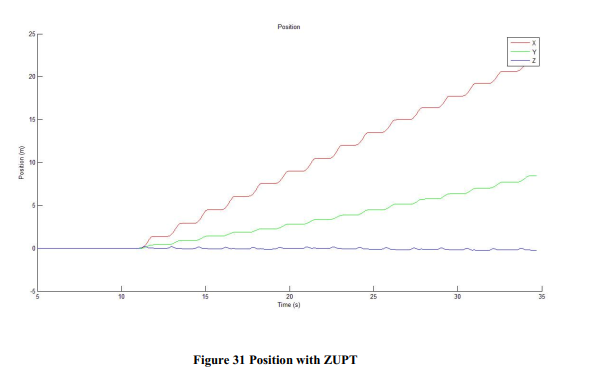


Figure 31 shows the position estimate of the 30 seconds of straight walk.

1. ENHANCED HEURISTIC DRIFT ELIMINATION
   1. Heuristic Drift Elimination

In order to remove bias and errors from MEMS gyroscopes, a method called  
Heuristic Drift Elimination (HDE) was proposed by Johann Borenstein and Lauro Ojeda in 2010 [32]. Like in zero-velocity update, when existing algorithms are not sufficient to correct drift, we make use of external facts to provide more information. ZUPT takes advantage of the fact that during walking, either of the feet undergo a repeating sequence of stop and move. HDE takes into consideration the fact that in most buildings, corridors  
are parallel or orthogonal to each other, HDE calls the directions of the corridors dominant directions. Therefore most of the time the walker walks in a straight line along one of the four dominant directions. In 2012, A.R. Jiménez et al. proposed a magnetically-aided HDE algorithm that works in more complex buildings [33]. These are the two most cited work in Heuristic Drift Elimination and are the latest advancements.

* 1. Enhanced Heuristic Drift Elimination in 3D

In this paper we propose a novel HDE algorithm that works in 3D and can  
completely remove gyro drift in structured, indoor environment. Compared to HDE by Johann Borenstein and Lauro Ojeda, our EHDE algorithm works on much cheaper MEMS IMUs that cost only a few dollars each while Johann used a sensor called Nano IMU produced by Memsense Inc. that costs more than $1300. HDE also has the limitation in that it can create new azimuth error by matching the closest dominant direction if the pedestrian walks in various directions. Compared to A.R. Jiménez’s work, our EHDE can work without the aid of a compass. Also, both of the two HDE algorithms mentioned above do 2D drift elimination only while ours corrects gyro drifts in 3D.  
Implementation of EHDE is explained below. EHDE is mainly composed of three parts: pedestrian motion detection, dominant direction calculation and position update

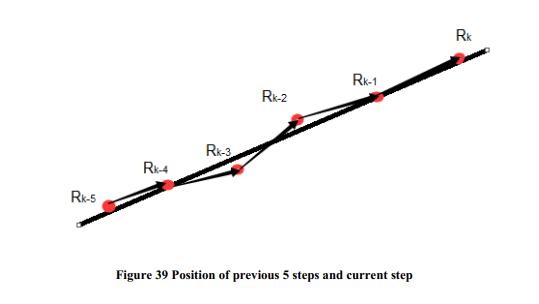
* 1. Pedestrian Motion Detection

In the beginning of drift elimination, EHDE takes the first five steps of the data and decides if the five steps are in a straight line. Since we use a low-cost MEMS IMU, gyroscope drifts really fast. So taking more steps would bring gyroscope errors in dominant direction calculation. Also large delays can appear. But if less steps were taken, motion detection is likely to be less accurate. According to our experiment, five steps provides a reliable and fairly accurate result.  
To decide whether the pedestrian is walking straight, we first calculate step  
vectors from position data. Then we take the first five steps and calculate the angles between adjacent two steps. If absolutes of all of the angles are less than a threshold, we say the pedestrian is walking straight. If not, the pedestrian is not walking in a straight line.  
If the pedestrian is not walking straight, EHDE does nothing.

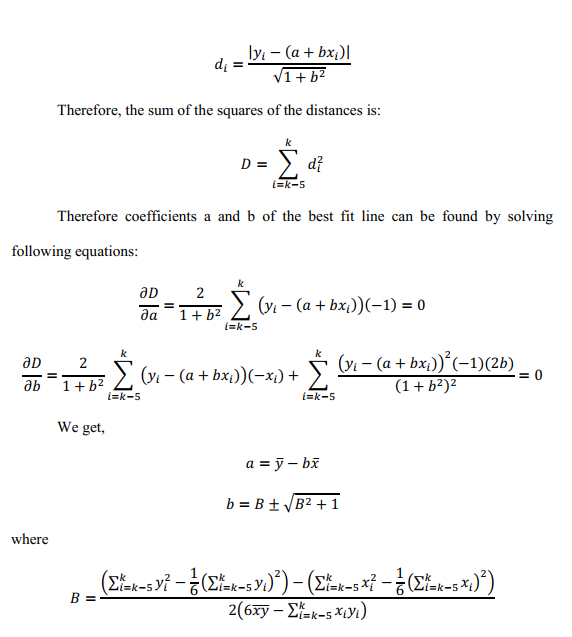
If the pedestrian is walking in a straight line, EHDE moves to the next step, which is the dominant direction calculation.

* 1. Dominant Direction Calculation

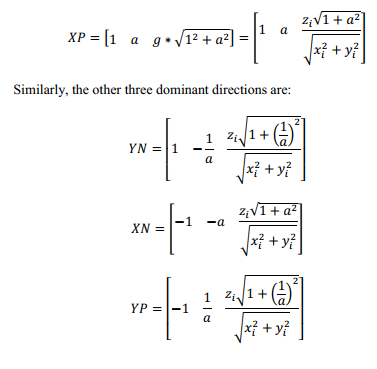
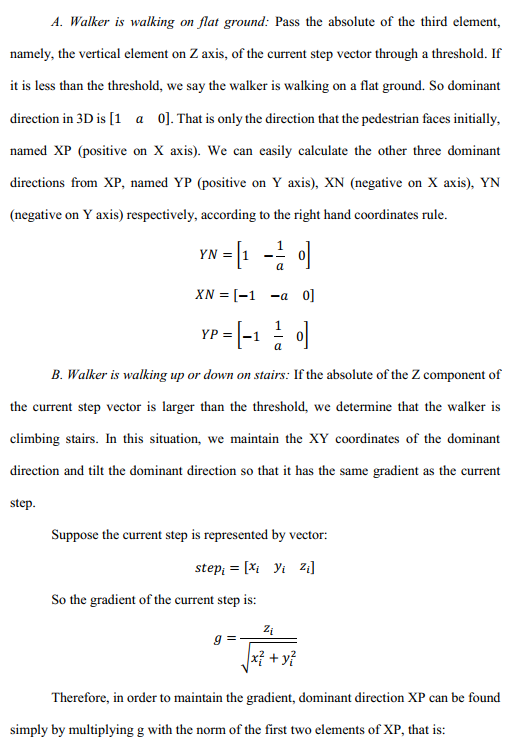
Dominant direction calculation consists of two steps: the first step is to calculate the dominant directions’ projections on X-Y plane. The second step is to generate dominant directions in 3D space.  
Dominant direction projection is calculated from the five steps mentioned above.  
Five step vectors are calculated from 6 positions. We perform linear regression which fits a straight line based on perpendicular offset through the projections of the first 6 positions on X-Y plane, as is shown in Figure 39.



We assume a straight line in X-Y plane with function: y = ax as the dominant  
direction projection.  
The distance from the points to the straight line in Figure 39 is given by:



Thus we get the dominant direction’s projection [1,a] on the X-Y plane.  
To calculate dominant direction in 3D space there are two situations that need to be treated differently. Because floors in buildings are generally connected only by stairs or elevators, there is no possibility that a pedestrian gradually descends or ascends.  
Consequently only two situations are possible: walking horizontally or climbing stairs.  
Determination for 3D dominant direction is explained below.

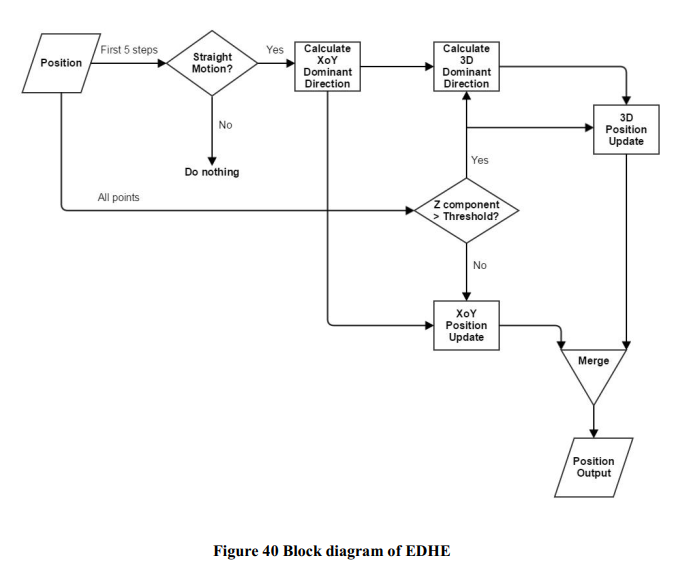


* 1. Position Update

The final step of EDHE is position update. In this step, first EHDE calculates the heading difference between the current step and each dominant direction and see if the heading difference is less than a threshold. The threshold is kept small so that there is no crossover zone of the possible range around each dominant direction. Therefore the current step either falls into the possible range of one of the four dominant directions or none of them.  
If the current step doesn’t belong to any possible range of the dominant directions, EHDE does nothing. If the current step falls in one of the four dominant directions, do the follow:

*Step 1:* Calculate the quaternion HDEquat between the current step and the  
dominant direction.  
*Step 2:* Use HDEquat to rotate all the position points after and including the  
current step to the dominant direction. The reason to rotate all the points is that because we use a low-cost MEMS IMU, it drifts at a rather fast rate. If only one step vector is corrected once, the following steps is likely to drift to the possible zone of another dominant direction and that would bring about destructive consequences to drift elimination. By rotating the whole remaining trajectory we can correct the drift little by little.  
*Step 3:* Because quaternion rotation is not a rotation around a certain, it is a  
rotation of certain degrees around a fixed axis. So after the rotation, even though the remaining trajectory is parallel to the previous step, they are not continuous. So the last step has to subtract the difference between the two ends of the “breakpoint” to make the trajectory continuous.

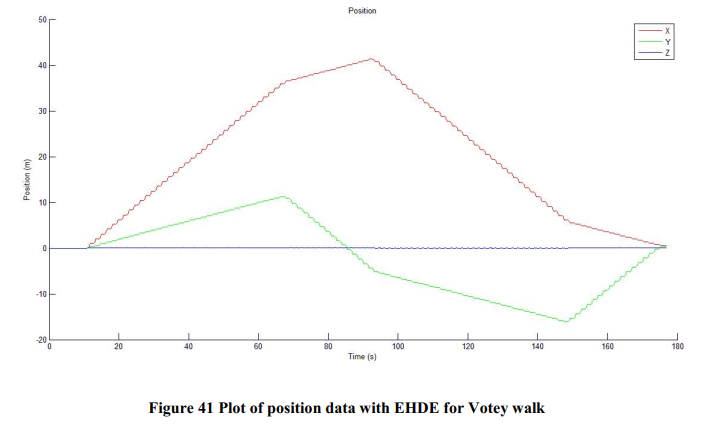
* 1. Block Diagram

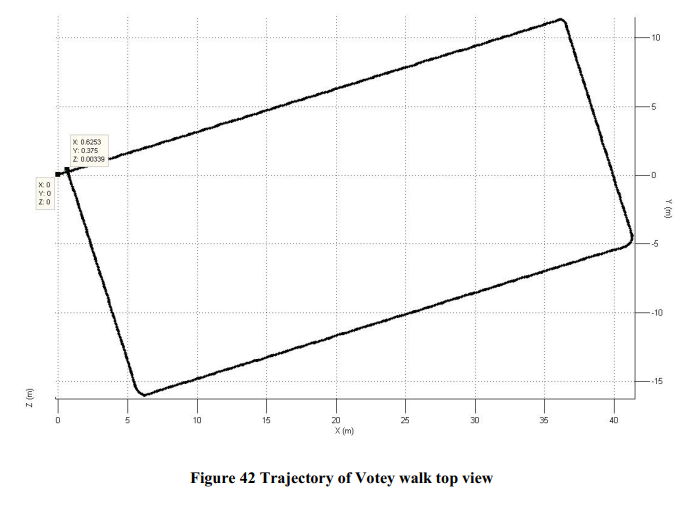


From the block diagram we can see that EHDE only takes in position data and  
outputs updated position data. It is independent of the AHRS algorithm.  
The reasons for maintaining dominant direction only on X and Y axes and keep same gradient on Z axis are as follows:  
First, floor height varies from building to building, even within the same building. Therefore the system cannot tell whether it is drift.  
Secondly, situation may vary according to how a pedestrian walks. Suppose the height of one stair is h. If pedestrian steps out his foot with the sensor first when he starts to go upstairs, the height change is, h, 2h, 2h, 2h…but if he steps out his foot without the sensor first, the height change would be, 2h, 2h, 2h, 2h…This makes height change uncertain and so it is difficult to reduce height drift.

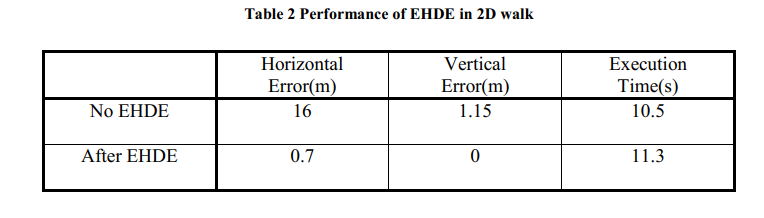
Thirdly, as we can see from the two sets of results in Section 5.1 Preliminary  
Results, drift on Z axis is negligible. Usually there are no more than 20 stairs between floors so if there is drift it is within an acceptable range.

* 1. Results





From Figure 42 we can see that the trajectory forms a perfect rectangle  
representing the corridor in Votey Hall. Walker started at [0, 0, 0]. The final point in position is [0.62, 0.37, 0],



From Table 2 we can vividly view the power of EHDE. Vertical drift is eliminated and horizontal drift is reduced significantly. At the same time, complexity of the algorithm does not increase much, EHDE trades off 99% of the error with less than 10% more execution time.