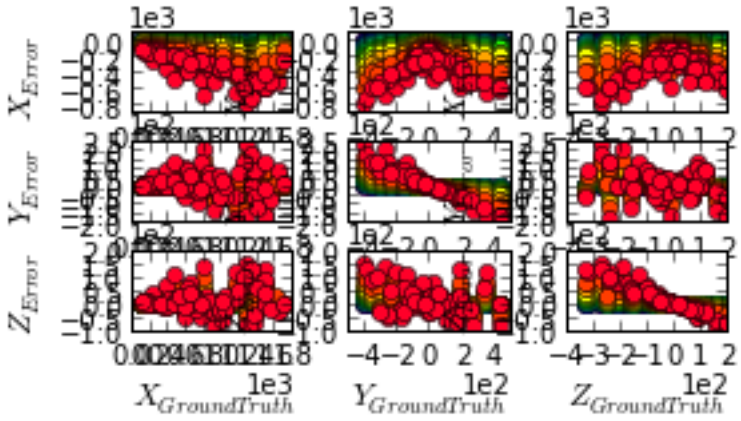


Figure 1: testing

Chapter 1

Introduction

Generating an accurate and high precision model of its surrounding environment to indicate hazard features is an important issue for any autonomous vehicle. Knowing its own location in the map is essentialfor the vehicle to navigate and avoidobstacle autonomously.

In many applications, the mobile robot has a priori map. The given priori map may be sufficient for localization purpose, but generally do not have the resolution required for obstacle detection. Ground vehicles need to deal with temporary added road block and parked cars. Arial vehicles may not have a high enough resolution map that indicates tall trees, steep hills or electrical towers. In addition, useable map do not always exist. Without maps or externally referenced pose information, the robot must produce its own map and concurrently localize itself within the map. This problem is referred to as the simultaneous localization and mapping (SLAM).

Traditional 2D SLAM algorithms are well established in the past decade [ref]. A SLAM algorithm typically utilises measurements from several types of sensor which can be divided into two groups, those that provide vehicles pose and orientation measurement, such as wheel odometry, GPS, or IMU; and those that provide landmark bearing and range, measurement, such as radar, sonar, laser range finder. In recent years, optical sensors are actively being incorporated into SLAM algorithm and

successfully used in ground vehicle navigation [ref]. For aerial vehicles, the experiments are mostly limited to simulation, and results with realistic aerial video data are unavailable.

1.1 Problem Statement

Obstacle detection has received a lot of research interest in recent years. Various algorithms were developed for ground, under water and aerial vehicle using different sensor such as sonar, radar, LIDAR, and vision. Most research focus on utilizing with only one sensor. Yet, research shows that using multiple sensors produces a better measurement than single sensor [reference in proposal]. With various sensors readily available on most UAV navigation hardware; such as accelerometers, gyroscope, GPS receiver, altimeter, etc., fully utilizing these sensors to aid the main OD sensor helps to improve the accuracy and robustness of the range measurement, especially in harsh flying condition.

This thesis focuses on developing anobstacle detection system by using a SLAM algorithm as a sensor fusion framework for medium size UAV conduction low altitude terrain following flight in natural environment.The obstacles are static objects on ground, and moving objects are not considered. Research presented in this thesis contributes to the project of developing a mid size UAV to perform geological survey, carried out by Carleton in collaborated with Sander Geophysis Ltd., an Ottawa based company specialized in high precision geophysical survey. To achieve high resolution data acquisition, the UAV must be able to perform terrain following flight with altitude from ground as low as 10 meters at speed ranging from 60 knots (30 m/s) to 100 knots (51.4 m/s). The rate of climb for the UAV is specified to 400ft of vertical rise per minutes (122 meters per minutes) [**?**] A quick analysis on the UAV specification and aerodynamic behavior reveals the requirement of a practical obstacle detection

system. Assuming a tree height of 20 meters, which is the average height for oak or pine, to allow for enough time to avoid obstacle, the UAV must be able to detect the threat at least 610 meters away from it (Figure 2). This analysis indicates that the obstacle detection must be able to map object up to a few thousands away from the UAV.

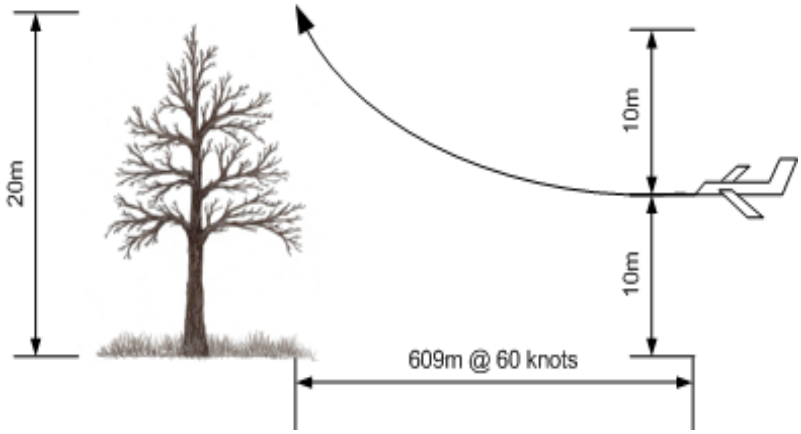


Figure 2: Case study for obstacle detection requirement

Although digital terrain map are generally available for flight path planning and in flight navigation, it does not have the resolution to indicate all hazardous obstacle such as tall trees, steep hills, or man-made tall objects. The obstacle detection and avoidance system must be in place to detect discrete threat, and operate automatically with minimum intervene from the operator.

1.2 Contributions

The thesis reviews the properties and consistency of a typical Extended Kalman Filter (EKF) based SLAM algorithm, and discusses the advantage and limitation of vision based SLAM method. The analysis motivates the development of an improved method by fusing multiple sensors into a mono vision EKF based SLAM framework.

Using a monocular vision for mapping is a bearing only SLAM problem. The measurement is through projection, which loses information about the relative position of the feature since the range is unknown. Without camera motion measurements, map created by monocular vision can be scaled arbitrarily. For a SLAM application in aerial scenario, camera vibration and sudden movement is common when aircraft is hit by cross wind, which can cause the lost of tracked features. A recursive EKF based SLAM algorithm is described in this thesis. The method utilizes sensor onboard the UAV to provide motion measurement of the camera, and improve the robustness of the algorithm under rough flying condition. Real aerial data were collected to test the performance and accuracy of the algorithm in a scenario similar to the one where the UAV will be eventually deployed. The preliminary result of the test flight was published in []. This paper is one of the first ones in the field that successfully applying monocular vision SLAM in large scale aerial application. A more thorough analysis on the behavior of the algorithm and its error is presented in this thesis.Furthermore, a number of baseline separations for the cameraare tested to optimize the performance and computation cost of the algorithm.

1.3 Organization

The thesis is organized as follows:

- Chapter 2 presents an overview on sensors, computer vision algorithms, SLAM algorithm related to obstacle detection and range measurement.
- Chapter 3 describes the detail implementation of the proposed multisensory monocular SLAM algorithm.
- Chapter 4 describes camera calibration procedure, equipments setup for the aerial data collection, and data preparation steps.

- Chapter 5 presents detail analysis on the performance of the algorithm. Convergence and consistency of the algorithm is discussed in section and . Error analysis compared to ground truth data is presented in and . The effect of using multiple sensors in improving the robustness is discussed in . At last, the camera baseline optimization is given in .

Chapter 2

REVIEW ON SENSORS AND RELATED WORK

2.1 SENSORS FOR OBSTACLE DETECTION

2.1.1 Overview

Many work in obstacle detection uses range sensors such as radar, laser range finder, LIDAR, sonar. [reference goes here]. Radar and laser range finder provides only point measurement at a given position and orientation. To acquire a full 3D range map of a scene, mechanical scanning mechanisms are required, which limits the data acquisition rate of these device. LIDAR operate in the same manner as laser range finder, except with the scanning mechanism built in. These sensors usually have high power requirement and mass, and may not be suitable for small and mid size UAV. Sonar is usually used in indoor or under water applications, and have wide beam profile which make it difficult to identify the origin of return signal, and results in low resolution range map. 3D flash LIDAR is capable of acquire 3D range measurement simultaneously by illuminating the entire field of view of the camera with a single laser, and capturing the reflected laser with a 2D imaging sensor [reference; wikipedia].

However, its high cost has limited its use in commercial application.

In recent years, many researches use optical sensor as a passive range sensor for its low weight, low cost. With the help of computer vision technology, optical sensors have been successfully used for range mapping and obstacle detection in a number of platforms [references]. There are several types of configuration in using optical sensor for range mapping: monocular, binocular, or multi-camera. Since optical sensors are bearing only sensors, the principle of range measurement is through triangulation a common scene point in two or more images captured. For binocular camera setups, two cameras are placed apart from each other with their relative geometry known and captures images simultaneously. If the position of a scene point can be accurately found in the images by both cameras, its distance can be calculated by using the difference in position of the projected point in images, and the separation of the cameras.

- Radar, sonar, laser range finder, 3D flash lidarvs. optical sensors
  - Radar, laser range finder have high power requirement and mass
  - Depth measurement can be obtained through optical sensors, which are inexpensive and light weight
  - Depth maps of a 3-D scene can be computed from a single pair of stereo camera. Stereo processing can require significant computational effort
- Monocular camera characteristic:
  - bearing-only sensor, which provide the measurement on the direction of the feature, and not the range. Other sensors, such as radar, are range and bearing sensors.

2.1.2 Monocular Vision and Binocular Vision

- Optical flow vs. feature detection and tracking
- The correspondence problem
- Initialization problem (addressed by Inverse depth parameterization)
- Lack of scale information of overall map -> must work with other sensors which provide robot motion measurement.

2.1.3 Limitation of Optical Sensor in Recursive Algorithm

- Error Accumulation over Iterations
  - Feature quality Decreases over Iterations

2.1.4 GPS and IMU

GPS and IMU are generally available on UAVs. These sensors provide a measurement on the robot motion. Odometry can provide the scale information which is missing in the bearing only measurement. Furthermore, odometry provides some prior information on the robot motion which can help to disambiguate the solution.

2.2 SLAM as A Sensor Fusion Framework

An essential aspect of autonomy for a mobile robot is the capability to determine its location. This capability is known as localization. Localization is typically a prerequisite for accomplishing real tasks, whether it is exploration, navigation toward a known goal, transportation of material, construction or site preparation. In many applications, the mobile robot has an a priori map. Given a map, the robot may localize by matching current sensor observations to features in the map. Given enough

features and an unambiguous geometry, the pose of the robot can be determined or at least narrowed down to a set of possible locations.

Usable maps do not always exist, and it is not always possible to have accurate externally referenced pose estimates. If an a priori map is not available, the robot may need to construct one. With a precise, externally referenced position estimate from GPS or similar means, the robot can take its sensor observations, reference the observations to its current pose, and insert features in the map in the appropriate places. Without maps or externally referenced pose information, the robot must produce its own map and concurrently localize within that map. This problem has been referred to as concurrent localization and mapping (CLM) and simultaneous localization and mapping (SLAM).

2.2.1 Recursive Probabilistic Estimation using Extended Kalman Filter

Present the typical EKF model for SLAM problem.

Properties of SLAM

Needs editing.Directly from papers.

Dissanayake proved three important convergency properties of the EKF solution to SLAM, namely that: (1) the determinant of any submatrix of the map covariance matrix decreases monotonically as observations are successively made; (2) in the limit as the number of observations increases, the landmark estimates become fully correlated and (3) in the limit the covariance associated with any single landmark location estimate reaches a lower bound determined only by the initial covariance in the vehicle location estimate at the time of the first sighting of the first landmark.

The properties imply:

- The entire structure of the SLAM problem critically depends on maintaining complete knowledge of the cross correlation between landmark estimates. Minimizing or ignoring cross correlations is precisely contrary to thestructure of the problem. (Early EKF for OD work eliminate the cross correlations between features and veichel pose in an attempt to reduce computation complexity.)

- As the vehicle progresses through the environment the errors in the estimates of any pair of landmarks become more and more correlated, and indeed never become less correlated.

- In the limit, the errors in the estimates of any pair of landmarks become fully correlated. This means that given the exact location of any one landmark, the location of any other landmark in the map can also be determined withabsolute certainty.

- As the map converges in the above manner, the error in the absolute location of every landmark (and thus the whole map) reaches a lower bound determined only by the error that existed when the first observation was made (Initialize the parameters using 1st frame as coordinate origin with minimum variance - Algorithm initialization).

It is important to note that these theoretical results only refer to the evolution of the covariance matrices computed by EKF in the ideal linear case. They overlook the fact that given that SLAM is a nonlinear problem, there is no guarantee that the computed covariance will match the actual estimation erros which is the true SLAM consistency issue.

Linearization Error and Consistency

Many research report filter divergence due to linearization error. Literature review here:

Huang investigate further on properties and consistency of nonlinear two-dimentional EKF based SLAM problem, and conclude:

- Most of the convergence properties in [3] are still true for the nonlinear case provided that the Jacobians used in the EKF equations are evaluated at the true states.

- The main reasons for inconsistency in EKF SLAM are due to (i) the violation of some fundamental constraints governing the relationship between various Jacobians when they are evaluated at the current state estimate, and (ii) the use of relative location information from robot to landmarks to update the absolute robot and landmark location estimates.

The robot orientation uncertainty plays an important role in both the EKF SLAM convergence and the possible inconsistency. In the limit, the inconsistency of EKF SLAM may cause the variance of the robot orientation estimate to be incorrectly reduced to zero.

Linearization error can be interpreted as error resulted from calculating the Jacobian at the estimated state (wrong state) instead of the true state.

Camera Centric Coordinate System

SLAM for Large Scale Maps

Chapter 3

Experiments with Real Data

3.1 Equipment Setup and Data Collection

In order to examine the accuracy and feasibility of the algorithm being used in low flying UAV for obstacle detection purpose, realistic aerial video and navigation data are collected through a test flight with the support of Sander Geophysics Ltd. A main purpose of the test flight is to obtain a piece of aerial video with the camera close to the ground as much as possible. This is difficult to achieve with any manned fixed wing aircraft. Therefore, a simulated unmanned aircraft system (SUAS) was used to carry all sensors. The SUAS is then towed by a helicopter via a tow rope of 33 meters long (Figure 3)to complete the survey. Yet, to prevent the SUAS from being caught by tree top, sufficient clearance must be left between the SUAS and the vegetation. As a result, the helicopter flew a planned path at approximately 100 meters above ground, and SUAS at approximately 70 meters above ground.

Sensors mounted on the SUAS included one wide angle CCD camera with 6 mm focal length lens capturing monocular image sequence at 30 fps, a pair of narrow angle CCD cameras for binocular images, one GPS antenna, and one flight control INS/GPS navigation unit Athena GS-111m [1](Figure 4). Analog video and navigation data are sent to the helicopter via three BNC cables and one data cable. Installed in the

Figure 3: Simulated UAS towed by helicopter

helicopter are two SGL data acquisition system CDAC. This system records video and data from SUAS, as well as data from sensors installed on the helicopter, including GPS, radar and laser altimeter, air pressure, temperature, humility, etc. Navigation data from the SUAS were sent in RS485, and were directly recorded via the UART port of the computer (figure 5). Videos from the three cameras were digitized to 720x480 resolution images using a PC/104+ MPEG4 video encoder from Parvus installed in CDAC. The video were time-stamped with GPS second on the image screen for post-flight synchronization with the navigation measurements. A snapshot of the digitized video is shown in figure 6.

Figure 4: Sensors mounted on SUAS. Top: the SUAS, bottom left: Athena GS-111m, bottom right: GPS antenna

3.2 Camera Calibration

3.3 Ground Truth Data Collection and Comparison

UAS localization ground truth are obtained through onboard flight control unit GS-111m. The unit records the SUAS position in GPS longitude and latitude coordinate. Orientation is obtained from the roll pitch and heading measurements. Roll and pitch accuracy has 0.1° mean and 0.1° standard deviation. Heading accuracy can achieve 0.5°. [1]

The estimated SUAS position can be directly obtained from the inverse of world origin estimation in the CC_EKF_SLAM state vector: $[O^c_{XYZ}, W^c_{XYZ}]$.

Digital elevation map (DEM) are downloaded from CGIAR-CSI website [2] and used as ground truth data for feature mapping. The downloaded DEM contains longitude, latitude and see level elevation of the terrain with a resolution approximately

Figure 5: Compact PCI data acquisition system (CDAC)

100 meters by 100 meters. To compare estimated feature position to the the DEM, some conversion are necessary to bring both data to the same coordinate system. In this work, the comparison are done in UTM coordinate.

First, the longitude and latitude in DEM data are converted into UTM using the WGS84 world geodetic system [3]. Many library are readily available to do the conversion by taking GPS coordinate and zone number as input. The library used in this work is a python interface to PROJ.4 library [4] called pyproj [4]. Secondly, the DEM data in UTM were converted into world frame using transformation matrix with intial SUAS position and orientation as input.

To bring result from the CC_EKF_SLAM algorithm to world frame, feature coordinates must be first converted into world frame using the estimated UAS localization results. Let $[X_i^W, Y_i^W, Z_i^W]^T$ be the feature coordinate in world frame, it can be calculated by

Figure 6: Image from monocular camera with GPS second timestamp

$$\begin{bmatrix} X_i^W \\ Y_i^W \\ Z_i^W \end{bmatrix} = Q^{-1}(O_{XYZ}^c, W_{XYZ}^c) \left(\begin{bmatrix} x_i^C \\ y_i^C \\ z_i^C \end{bmatrix} + \frac{1}{\rho_i} m(\varphi_i^C, \theta_i^C) \right) \tag{1}$$

Chapter 4

Error Analysis via Simulation

There are many factors impacts the accuracy in UAS localization and feature position estimation, and they can be sorted into three main categories:

1. Noise in system intrinsic parameters. The system intrinsic parameters includes
   - Camera intrinsic parameters
     - Coordinate of optical center on image plane $[c_x, c_y]$
     - Scaling factor to project feature in 3D world to image plane $[f_x, f_y]$
     - Lens distortion parameters $[k_1, k_2, p_1, p_2]$
   - Image resoluttion.
   - Accelerometer bias
2. Error introduced by LK tracking algorithm. Reliable vision tracking is an entire field of research in itself. Pyramid implementation of Lucas-Kanade (LK) tracking is used in this work, and there are a number of factors contribute to its performance. Firstly, LK tracking algorithm tracks features by comparing the intensity of a window of the image centered at the feature coordinate in image from one frame to another. The searching process terminates when the sum of error on the windowed image intensity is lower than a value set by user, or

when iteration of search has reached a maximum number set by user. Secondly, as scene evolve from frame to frame, the initial feature appears differently from frame to frame as the viewing distance and angle is different. Thirdly, sudden intensity change in the image sequences significant noise in the tracking. In outdoor setting, intensity change can be introduced by many factors, such as changes of sky area in a image, sun glare, UAV enters or exits cloud shades, or camera auto-adjust its shuttle speed, etc.

3. Error caused by the SLAM algorithm itself. The algorithm estimated features coordinate through a model that represents the relation between UAS location, feature location and UAS motion. As the model is non-linear, the linearization process introduces error into the result.

To better understand the impact of the factors listed above. A simulation is performed to examine the impact item 1 and item 3.

The simulator first generates a 3D point cloud ranging from 100 meters to 3000 meters from the camera. At each frame, the coordinates of the 3D points are first transformed to the new camera frame using the measured UAS motion. Next, the 3D points are projected to the image plane using a camera model defined by $[c_x, c_y, f_x, f_y, k_1, k_2, p_1, p_2]$, and digitized to a resolution of choice.

4.1 An Ideal Case

First of all, understanding the algorithm's performance under no noise, or nearly no noise condition provide a solid ground for the analysis later on. This simulation shows how much error the model itself generates under the most basic flying condition, which is moving forward at constant speed. The low noise environment is configure as such,

- UAS is moving forward (X axis) with constant speed at 60 knots.

- Y axis and Z axis translational movement are limited to white noise with standard deviation of 0.08 meters and a mean of 0.
- UAS rotation are modelled by white noise with standard deviation of 0.01 degree and a mean of 0
- No error was introduced due to image digitization. (i.e. the projected feature position on image plane was not digitized to any sensor resolution)
- No error was introduced from camera model mismatch. (i.e. camera model used by simulator is exactly the same as the one used by CC_EKF_SLAM algorithm.

4.1.1 UAS Localization

The estimation of UAS coordinate and orientation is first analyzed, as these estimates are directly used to perform transformation between camera and world frame. Figure 7 plots the UAS translation and rotation against video frame number. The ground truth and estimated value are plotted in blue and green lines respectively. The error defined by $Estimated - GroundTruth$ is plotted in red line. Under a simple forward only motion, the algorithm tracked the UAS status quite well, with error on translational motion less than 1 cm and error on rotational motion less than 3e-3 degree.

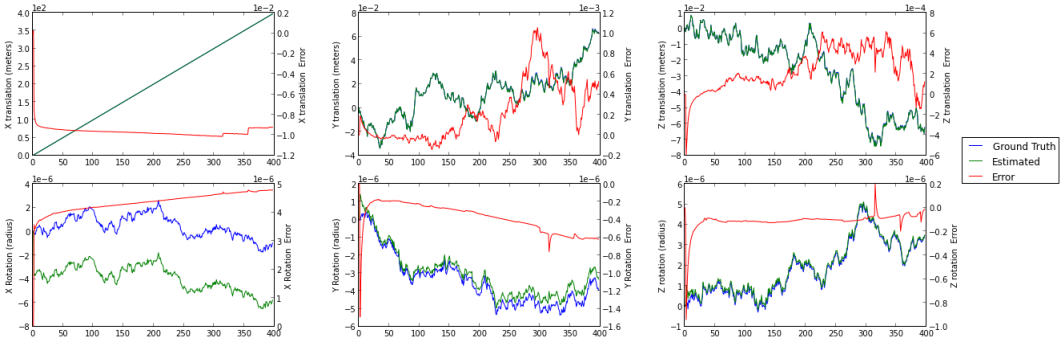


Figure 7: UAS localization error under no noise condition

4.1.2 Features Mapping - Convergence and Accuracy

Figure 8 left shows feature parameters $[d, \varphi, \theta]$ (where $d = 1/\rho$) plotted against frame number for the first 50 frames. The feature depth d for all features converged within 3 frames; elevation-azimuth angles $[\varphi, \theta]$ stay almost constant after initialization. A more detail graph can be seen from the error convergence plot for these parameters (Figure 8 right.) which shows the tracking error of these parameters for 400 frames. The error of feature distance$d$ continues to approach zero as the tracking continues. $[\varphi, \theta]$ show small drift within $+/-0.0002^\circ$ respectively. However, as tracking continue into later frames, error of $[\varphi, \theta]$ gradually grew bigger. The resulting error for feature coordinate in world frame represented in standard Euclidean XYZ parameterization is plotted in ??. The features positions errors in world frame converge to zero as tracking continues. During the process, certain features moved out of the FOV, and therefore its position estimation remained unchanged since then. At the end of the 400 frames, the x axis position error of the feature reduced to +/-0.2 meters; y and z axis error reduce to +/-0.02 meters

4.2 Effect of UAS Motion

The simulation result from UAS forward travel shows that the CC_EKF_SLAM algorithm does feature tracking and self localization quite well under simple UAS motion. Next, the algorithm is tested with a more complex and realistic scenario. A series of motion is added to the simulation in addition to the forward motion. The remaining 5 types of maneuver are added one at a time. These maneuvers are: translation on Y, translation on Z, rotation on X, rotation on Y and rotation on Z. Each motion is modelled by a sine wave with frequency at 1Hz, and variable amplitude. For translation on Y and Z axis, the sine amplitude varies from 1 meter to 19 meters with 2 meters increments. For X, Y, and Z axis rotation, the amplitude varies from 0.001

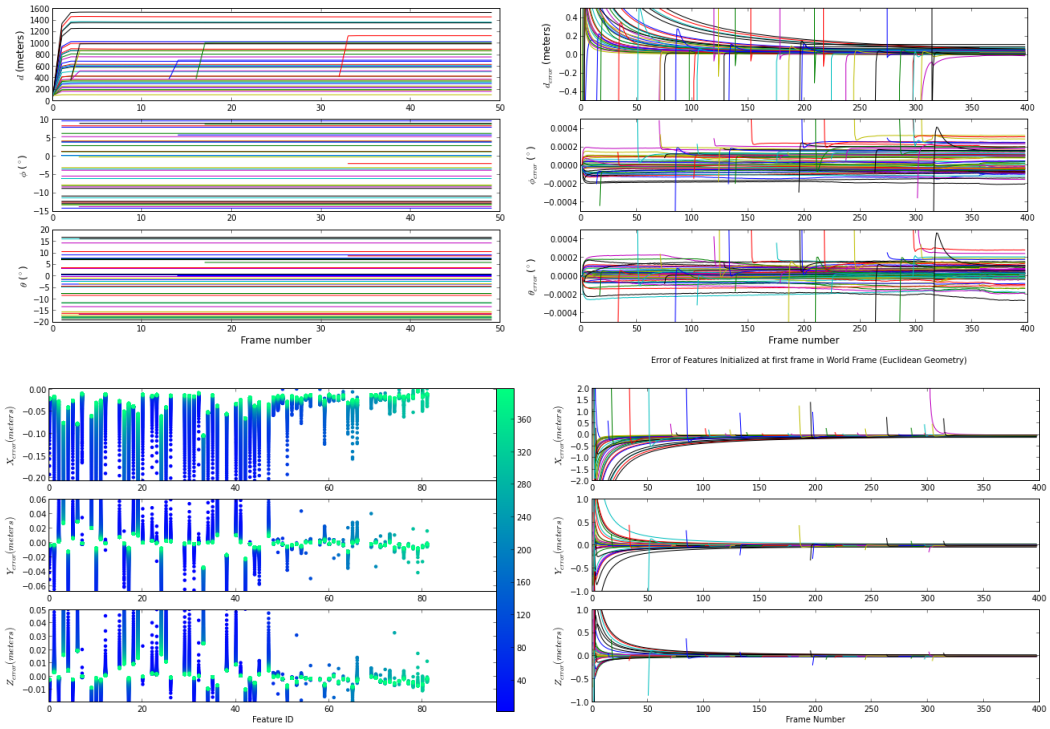


Figure 8: Features parameters and error convergence under no noise condition

radius to 0.018 radius with 0.001 radius increment.

4.2.1 UAS Localization under Motion

Figure 9 shows the UAS localization error statistic under translation motion on Y and Z axis. Figure 10. shows the UAS localization error statistic under rotation motion on X, Y, and Z axis. The blue dots mark the mean value μ of the error throughout the tracking, and the error bars mark the standard deviation σ.

The translation motion clearly increases the error of UAS localization. However, the amount of increase is insignificant. With the Sine amplitude increased to 19m, UAS position error increased by less than 0.02 meter.

On the other hand, rotation motions have a big impact on the accuracy of localization. Rotation on X axis has small effect on the accuracy of UAS position and orientation estimate. No obvious increase on mean and standard deviation of the

error can be observed. Rotations on Y and Z axis yield significant error increase on the position of the UAS.

- Rotation on Y axis increases the UAS X position mean error, as well as the standard deviation. For Z positioning of the UAS, the mean error stays zero, but standard deviation increase dramatically.
- Same thing happens to the X and Y positioning for rotation on Z axis.

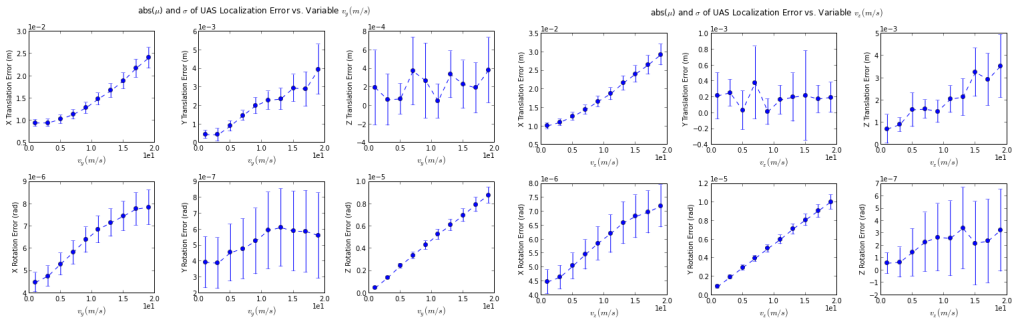


Figure 9: UAS localization error under translational motion

To understand how rotation motion affect the UAS localization estimation, the UAS position on X, Y, Z in world frame are plotted below (figure 11) with rotation amplitude on X, Y and Z set at 0.01 radius. With rotation on X axis, the position error of UAS shows some oscillation. The oscillation magnitude remains small (in the scale of millimeters) and around zero. For rotation on Y and Z, the situation is very different. Both rotation motions caused the UAS position error to oscillate with the oscillation amplitude increasing (diverging) as tracking goes on. The X position error of the UAS increases in positive value with both rotation on Y and Z, but the most significant impact happens on the Z position (for rotation on Y) and Y position (for rotation on Z), with error reaching 20 meters at the end of the video sequence. With rotation rate increases (amplitude of the sine wave), the rate of error diverging from zero also increases, hence, resulting in an increasing error standard deviation in

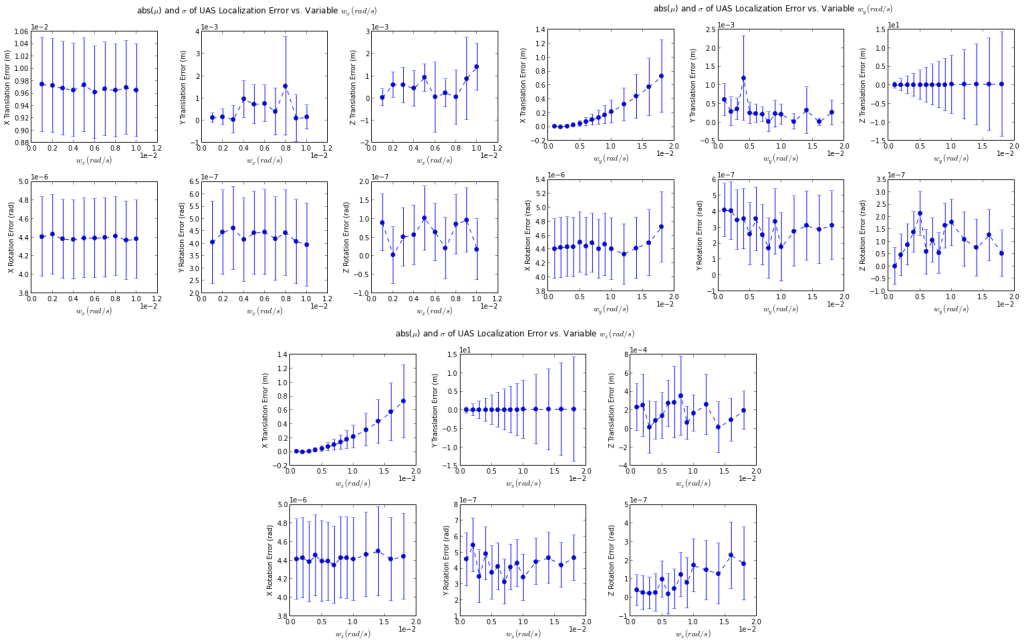


Figure 10: UAS localization error under rotatioanl motion

error statistic plots. This simulation result suggests that CC_EKF_SLAM algorithm is very sensitive to rotation on Y and Z axis.

4.2.2 Feature Mapping Accuracy under Motion

Feature mapping error statistic are drawn from the feature error at last frame, since features error converge to zero as tracking goes on. Figure 12 shows the feature mapping error statistic with translation motion on Y and Z axis. Translation motions increase both the error mean and standard deviation, but not by much. With the motion maximum amplitude ranging from 1 meters to 19 meters, the increases of features position error mean and standard deviation are both in the scale of centimeters.

Figure 13. shows the feature mapping error statistic for rotation motion on X, Y and Z axis. With maximum rate of rotation ranging from 0.001 rad/frame to 0.018 rad/frame

- Rotation on all three axis yields significant error increase for feature position

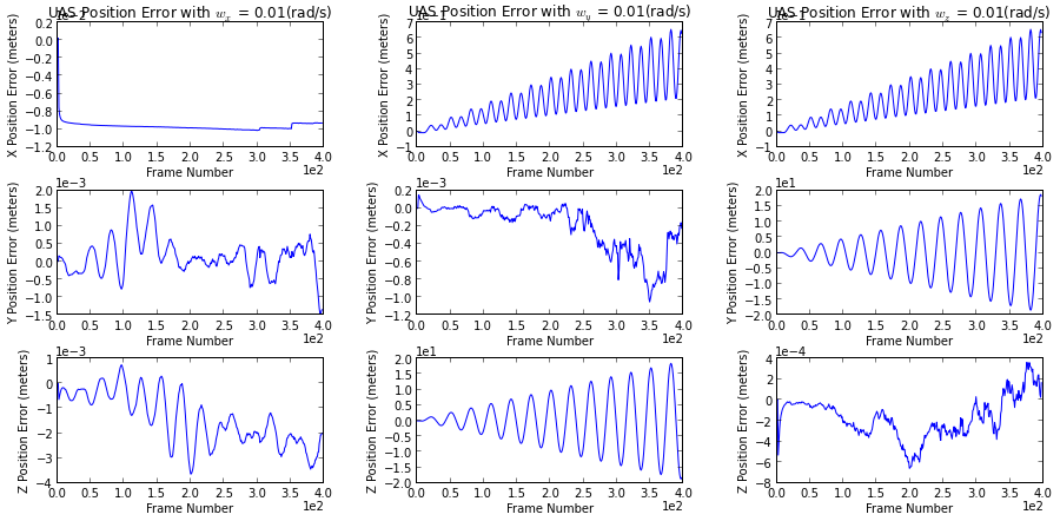


Figure 11: UAS estimated position in world frame

estimation

- X axis rotation causes increase on standard deviation of Y and Z axis feature position error, and by similar amount. The error are in scale of meters with maximum rotation setting.

- Y axis rotation causes increase on mean and standard deviation of X and Z axis feature position error. Z axis feature position receives the biggest impact with error in the scale of hundreds of meters with maximum rotation setting

- Z axis rotation impacts on X and Y axis feature position in a similar way as the Y axis rotation.

Figure 14 shows the features position error at the last tracking frame for all three type of rotation motion. The errors are plotted against feature ID and it revealed some more characteristic of the features mapping errors.

- With rotation on Y and Z, the tracked features easily went out of FOV. This can be observed from total number of features went from 80 to over 110 with Y

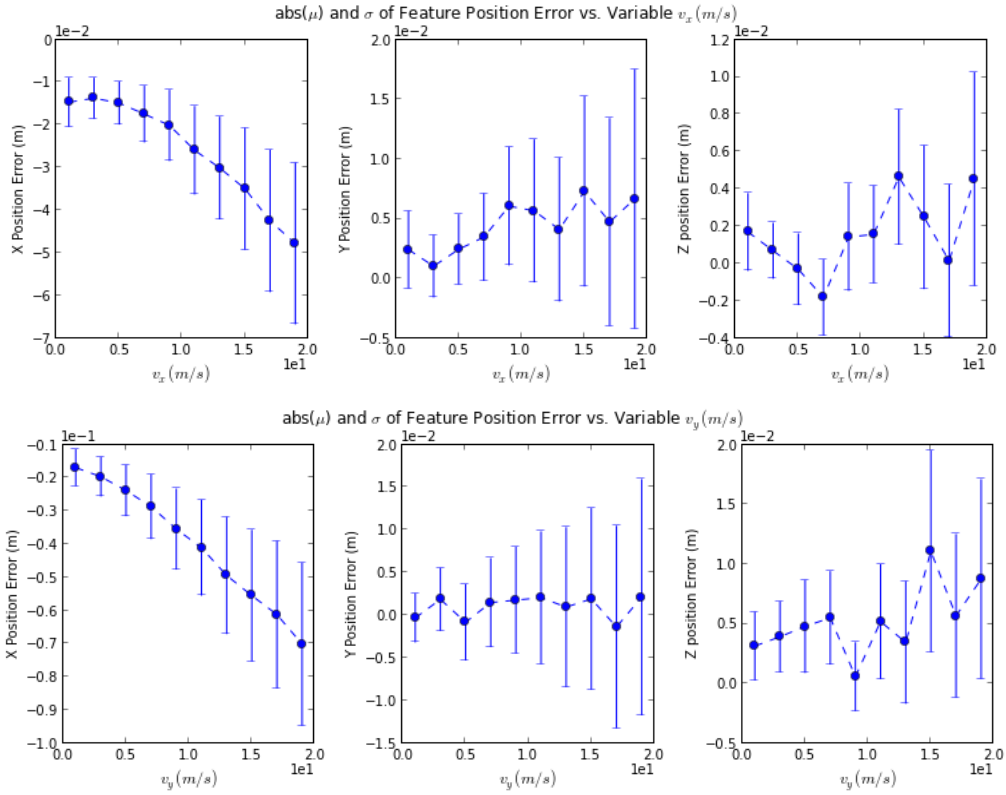


Figure 12: Feature mapping error under rotatioanl motion

axis rotation setting varied from 0.001rad/s to 0.018rad/s. This caused frequent addition of new features.

- Features added after first frame has much bigger error than features added at first frame. At 1^{st} frame, 40 features added to the filter. Error plots from Y and Z axis rotation both shows that major feature mapping error came from features added after the 1 st frame with ID bigger than 40.

To investigate how does rotation motion results in bigger error on features added after first frame, feature parameters error (converted to world frame) with $wy = 0.01$ are plotted in figure 15. It is found that the most significant error happen to parameter ϕ which is the feature elevation angle. This angle has the same definition as rotation angle around Y axis. The second contributor is z_i (the Z axis coordinate of the feature

initialization point). The both parameters have an offset error at initialization, and were never corrected throughout the tracking.

Figure 16 shows the ϕ error at initialization in camera frame and world frame. The blue line shows the error in camera frame. The red line shows the error in world frame transformed using the estimated UAS position and orientation. It is clear that it is the transformation process that introduced the offset error in ϕ. Offset error in z_i is due to the same reason. Recall that with Y axis rotation, UAS localization estimate has biggest error in X and Z axis coordinates estimate. Features initialized at first frame don't carry any offset error is because the transformation process is using the same parameters in both way. During tracking, these features are transformed to the new camera frame using the estimated UAS position and orientation. These are the same estimation being used to transform feature position from camera frame into world frame. Therefore, although the UAS localization estimations are different from the ground truth, feature initialized at first frame were not affected. To conclude, the major contributor for feature mapping error came from error in UAS localization estimation.

4.3 Camera Intrinsic Parameters

This section summarizes the impact of inaccurate camera parameters estimations. Error on camera intrinsic parameters is simulated by using different values for camera models. One model is used in the simulator to project 3D points onto image plane, and the other is used in the measurement model of CC-LK-SLAM. c_x, c_y, f_x, and f_y are simulated individually and distortion parameters $[k1, k2, p1, p2]$ are simulated as a group. Using the calibrated camera model (see section ???) as a base model,c_x, c_y, f_x, and f_y in simulator camera model varied from -50% to 50% of the base model. Distortion parameters varied from 0% to 140% of the base model.

4.3.1 Effect from (c_x, c_y)

Figure 17 show an over view of UAS localization error statistics with incorrect estimates of (c_x, c_y).

UAS position error is dependent on (c_x, c_y) and time (figure 18). The UAS position error is diverging (increases in time), and can be modeled by 1 st order polynomial function, with the rate of diverging decided by the error of (c_x, c_y). c_x affects UAS position on X and Y axis, and c_y affects UAS position on X and Z axis.

The feature mapping error statistics from incorrect estimate of (c_x, c_y) are plotted in figure 19. The following characters can be observed from the plots:

- Incorrect c_x affect feature position on all axis, among which, X and Y axis see the most significant error.
  - The further c_x deviate from the true value, the further the features appear (positive X axis error).
  - On Y axis, smaller c_x make feature appear further to the optical axis than ground truth (positive error), and bigger c_x make features appear closer (negative error).
  - Incorrect c_x also affect Z axis feature position, but by a much smaller amount.
- Incorrect c_y affect feature position on all axis similarly to c_x. Estimates on X and Z axis show most amount of error.

Plotting feature position error as a function of features ground truth positions reveals more information on how incorrect c_x and c_y affect feature mapping. Feature position error is a function of its ground truth position, and c_x (or c_y).

- Error on X axis is proportional to the ground truth position on X. The further the feature is, greater the error. The degree of incorrectness in c_x decide the

slope of the error plot, greater the error in c_x, steeper the slope (figure 20, plot (a), subplot $[1, 1]$).

- Feature position error on Y axis is also proportional to the ground truth position on X with the slope polarity dependent on the polarity of the error of c_x, and error plot slope dependent on the error of c_x.

- Feature position error on Z axis is proportional to the ground truth position on Z, with slope polarity dependent on the polarity of error of c_x, and slope dependent on the error of c_x.

c_y affects feature position similarly to c_x (figure 20, plot (b)), except feature position error on Z is dependent on ground truth position on X, and error on Y is dependent on the ground truth position on Y.

4.3.2 Effect from (f_x, f_y)

With (f_x, f_y) varying from -50% to +50% of the calibrated value, the UAS localization error is shown in 21. For all f_x and f_y settings, UAS position error remained less than +/-0.05 meters, and orientation error remained in less than 8e-6 radius. Compared to the error obtained from the ideal case simulation, Error in (f_x, f_y) estimation does not introduce any additional error into UAS localization estimate.

Feature mapping, on the other hand, is unavoidly affected by the error in (f_x, f_y) since these are the scaling factor that project features from 3D world onto image plane. Figure 23 shows the error statistic of feature position estimation under variation of (f_x, f_y). The effect on the X axis component is minimum, in the scale of milli meter. Y axis component receive the most impact with unmatched f_x, since this is the scale factor that map feature's Y component in world frame onto U axis on image plane by $u = Y/X * f_x$. Same goes for the Z axis component of the feature position estimate and its relation to f_y.

Plotting the feature mapping error against feature ground truth position reveals how error in (f_x, f_y) estimate impact on feature position estimate. When f_x estimate contains error, Y component of feature position is determined by the feature's ground truth position on X and Y components. The value of Y_{error} is directly portional to the Y component of its ground trurth position, with the error in f_x determines the function's slope. Same relation can be found for Z_{error} with f_y, and $Z_{groundtruth}$.

4.3.3 Effect from Distortion

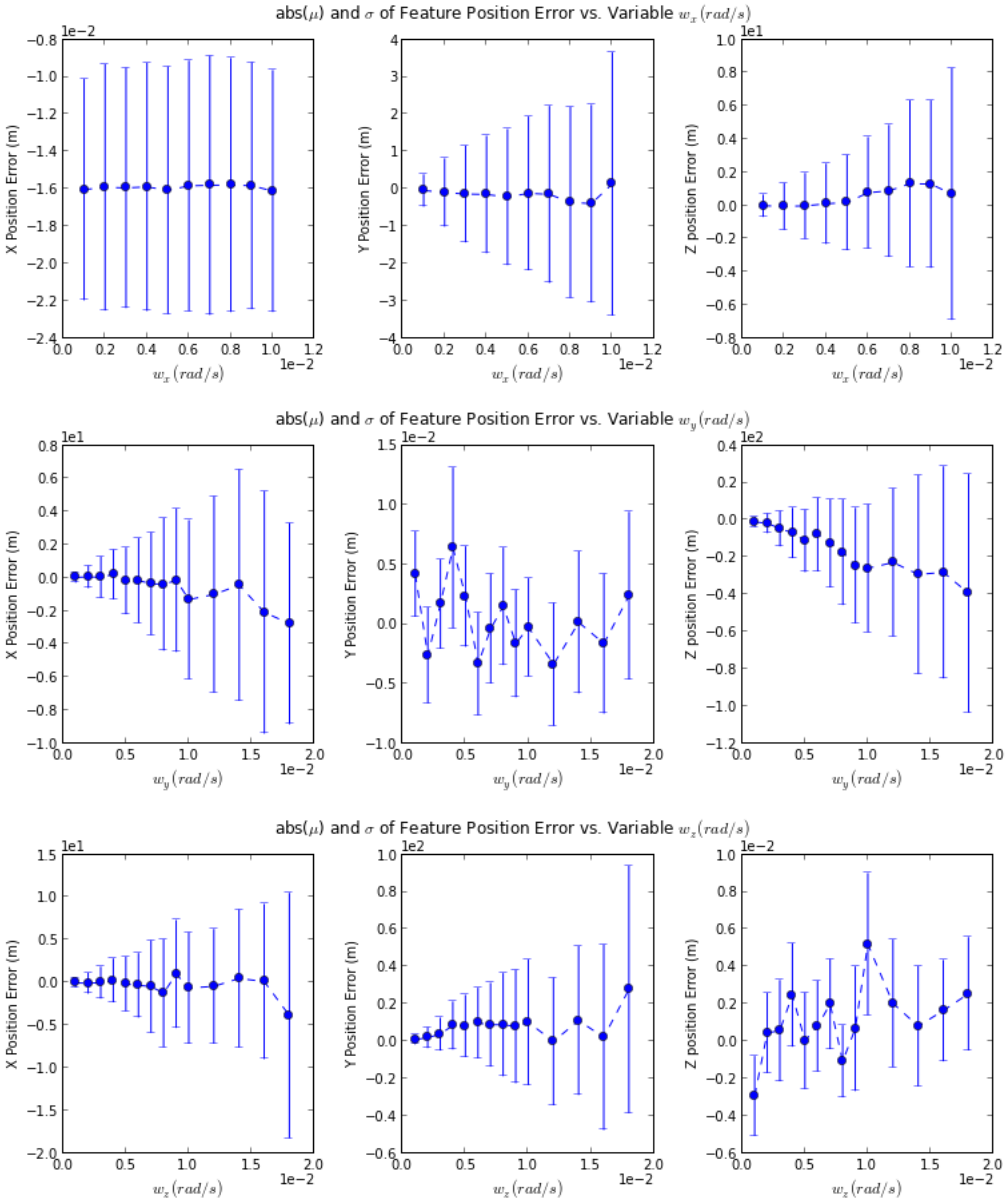
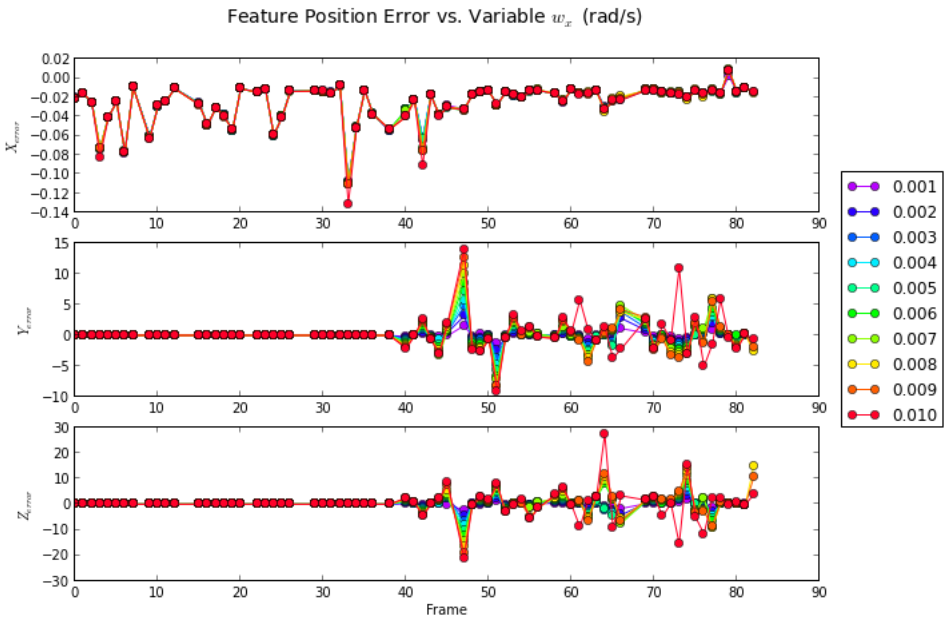
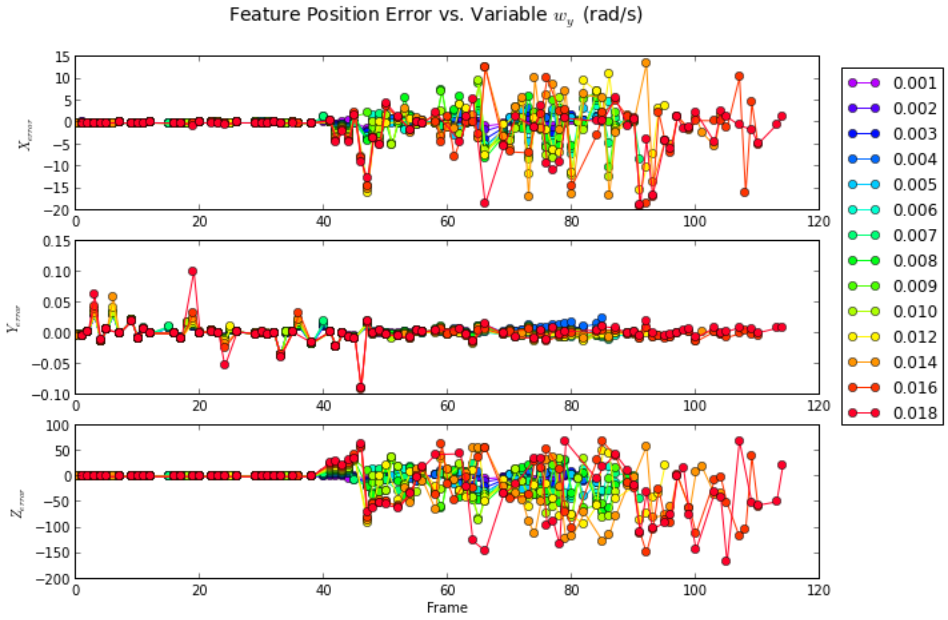
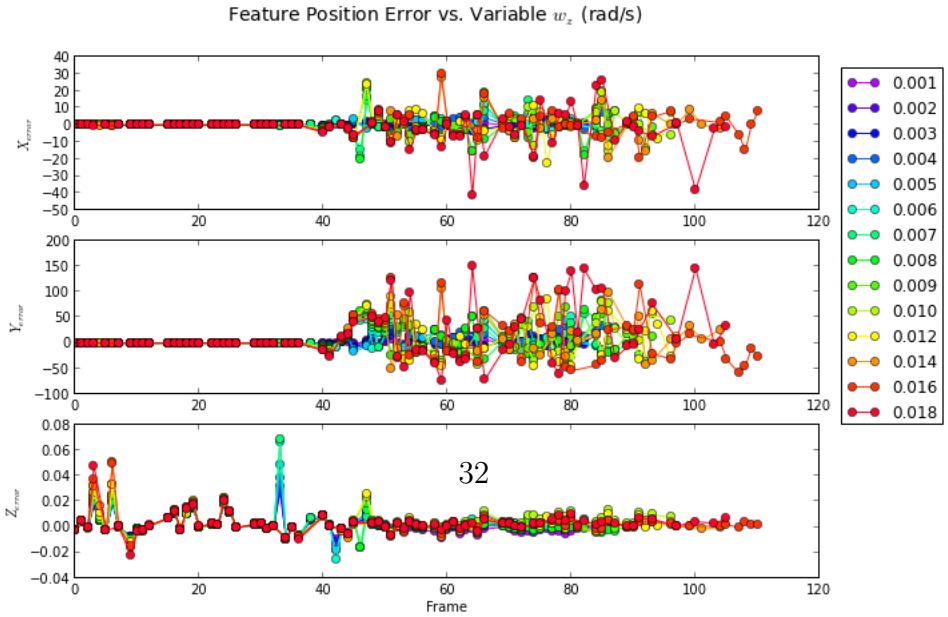


Figure 13: Feature mapping error under rotatioanl motion


Feature Position Error vs. Variable w_x (rad/s)
X_{error}
Y_{error}
Z_{error}
Frame
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.010


Feature Position Error vs. Variable w_y (rad/s)
X_{error}
Y_{error}
Z_{error}
Frame
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.010
0.012
0.014
0.016
0.018


Feature Position Error vs. Variable w_z (rad/s)
X_{error}
Y_{error}
Z_{error}
Frame
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.010
0.012
0.014
0.016
0.018

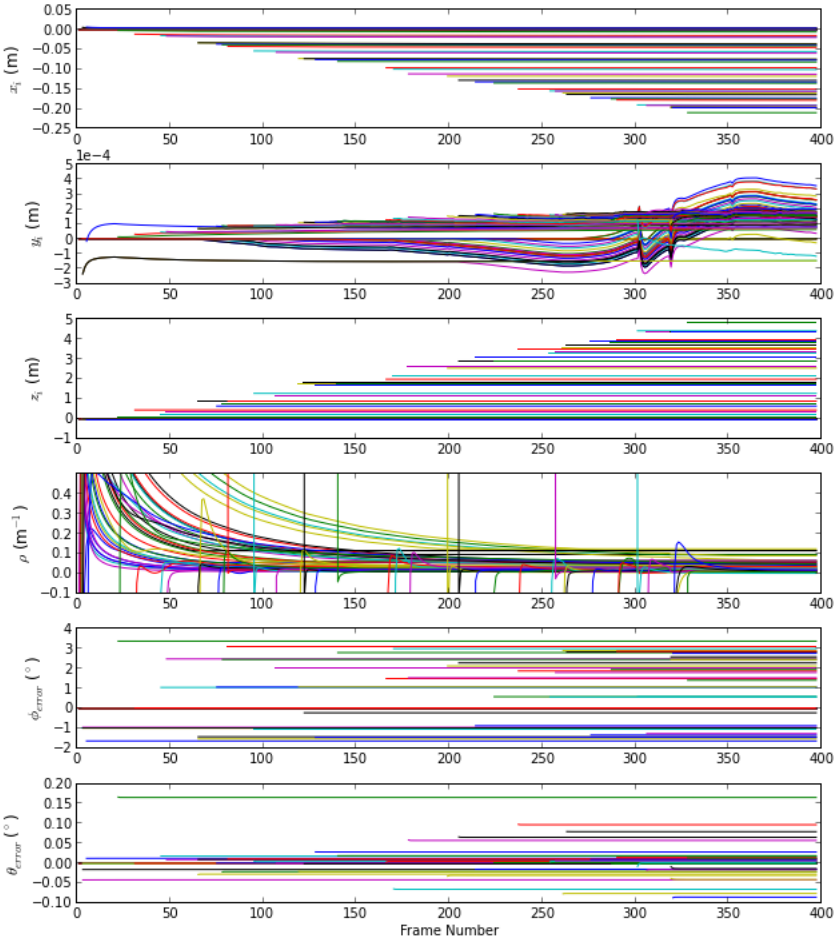


Figure 15: Feature parameters error under rotatioanl motion

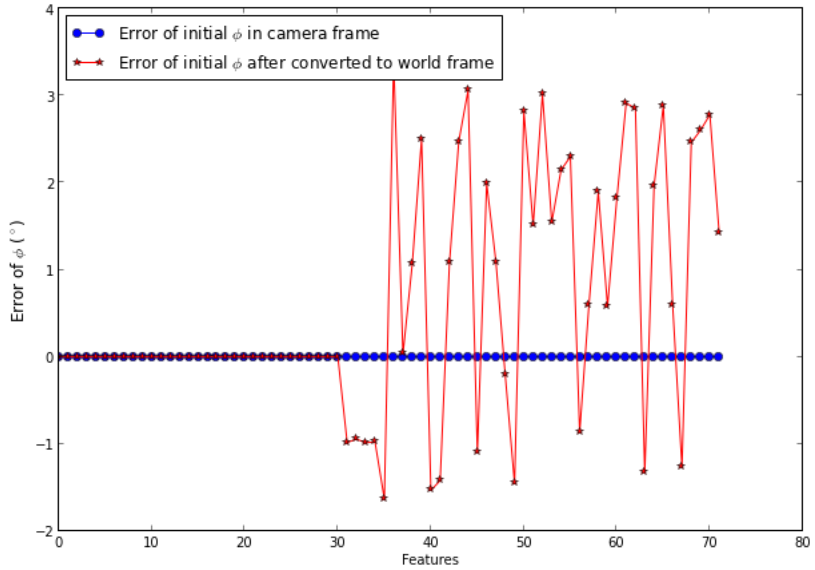


Figure 16: Error of ϕ in camera frame and world frame at initialization

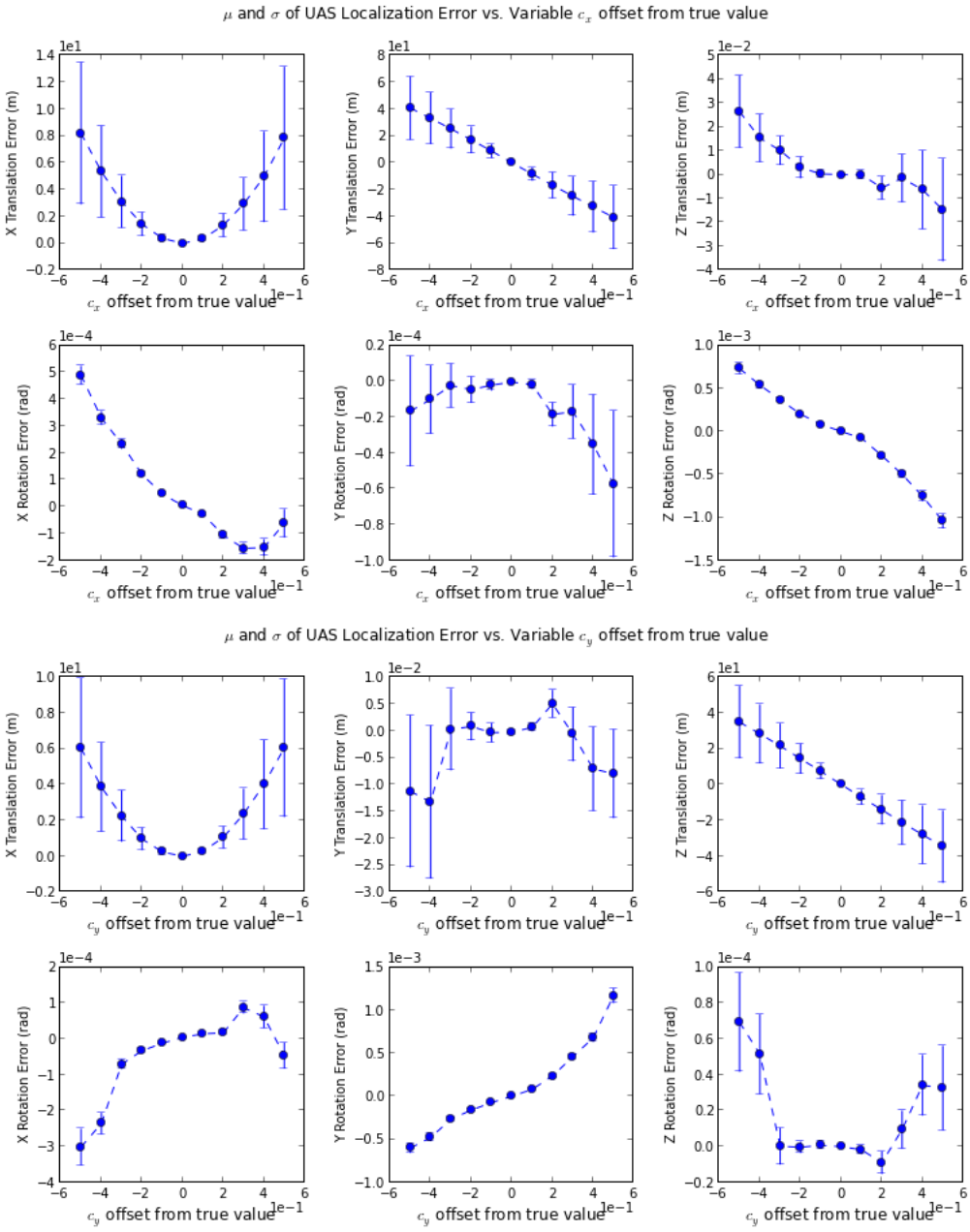


Figure 17: UAS localization error statistic with varying (c_x, c_y)

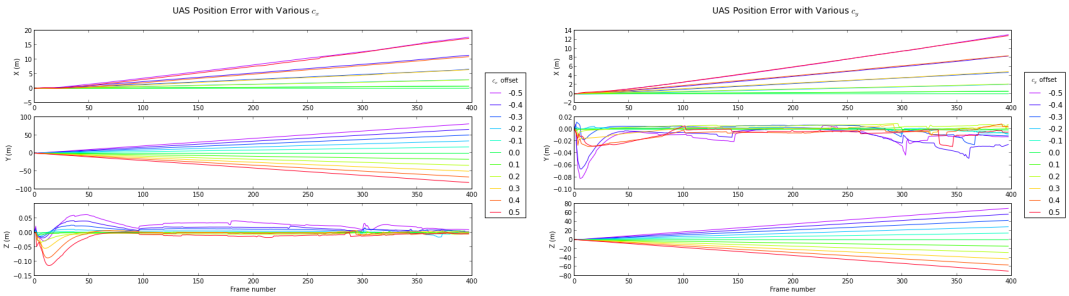


Figure 18: Diverging UAS position error

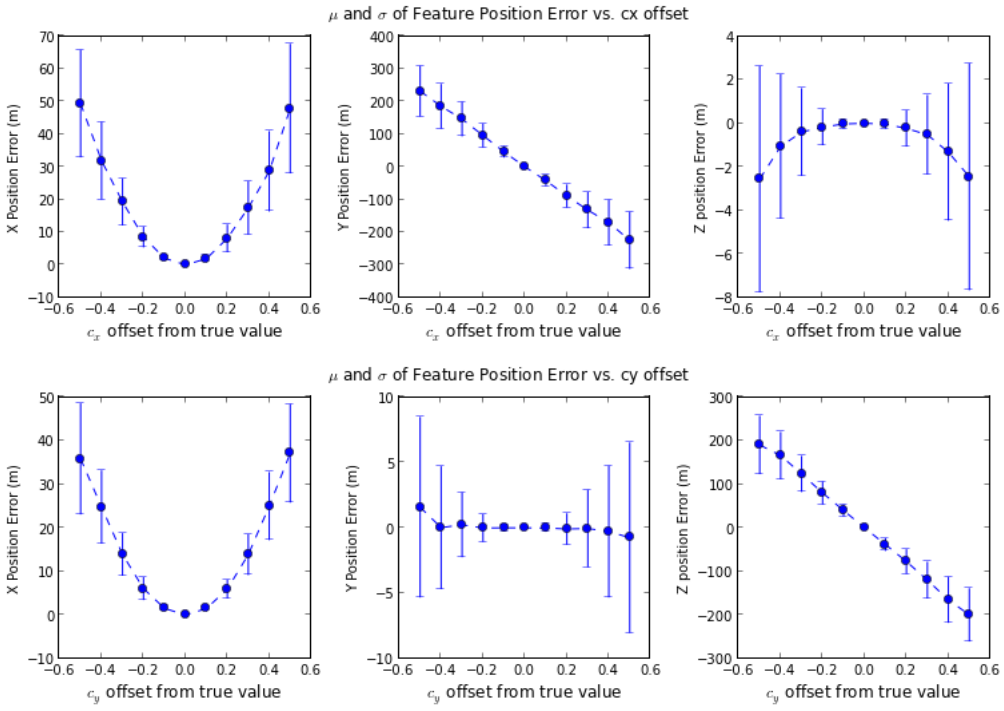


Figure 19: Feature mapping error statistic with varying (c_x, c_y)

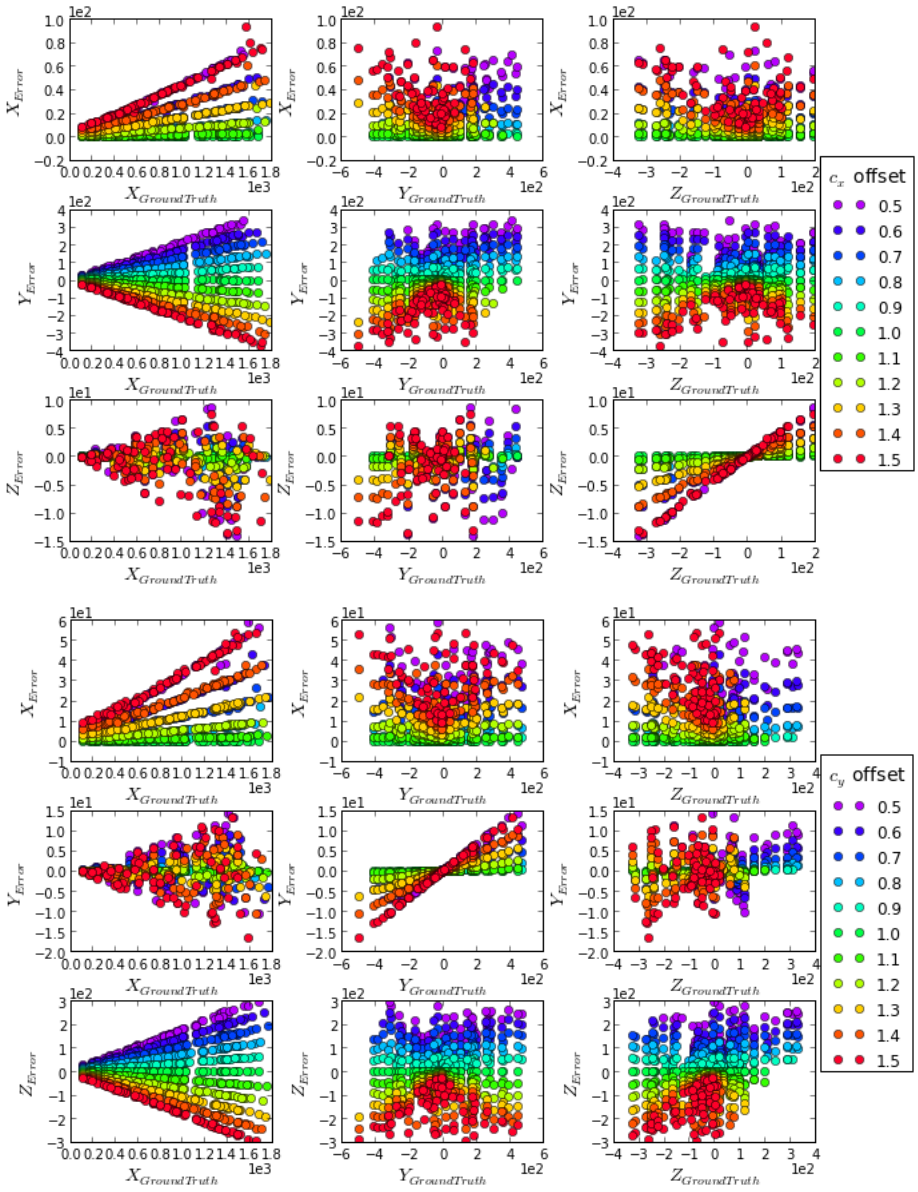


Figure 20: Feature mapping error vs. ground truth feature position

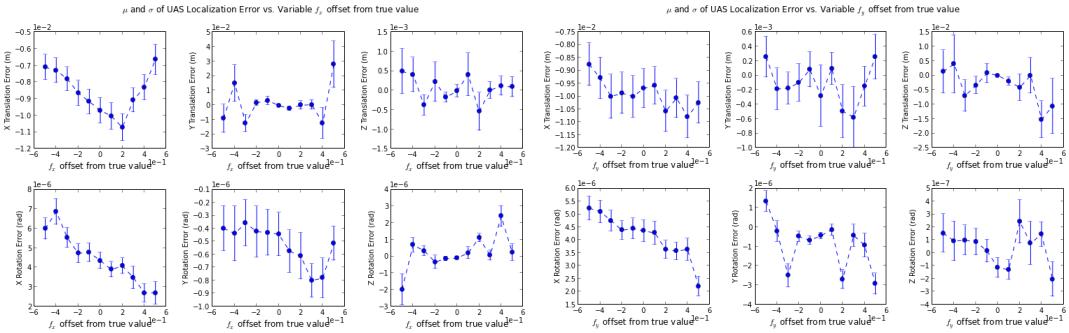


Figure 21: UAS localization error statistic with variable (f_x, f_y)

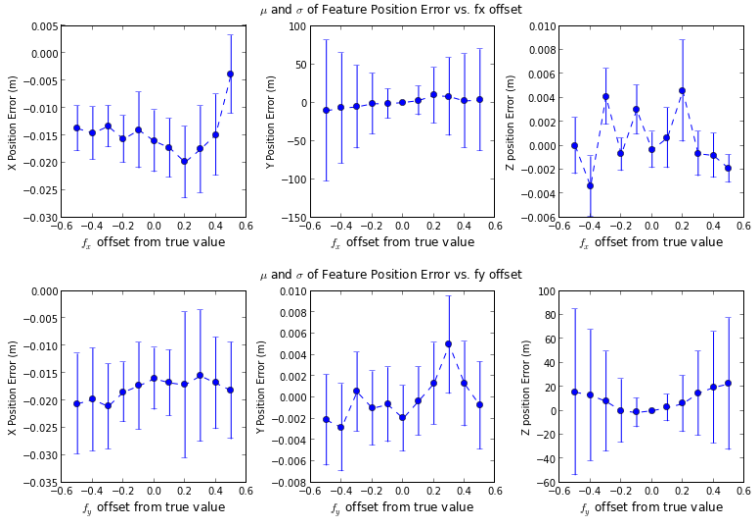


Figure 22: Feature mapping error statistic with variable (f_x, f_y)

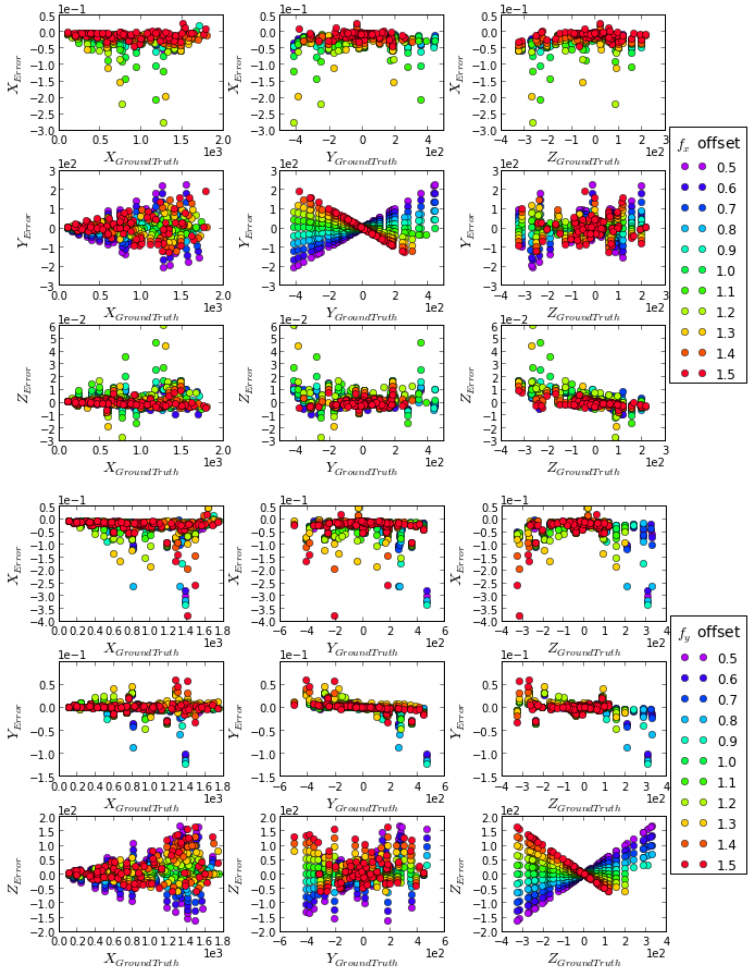


Figure 23: Feature mapping error plotted against feature ground truth for various (f_x, f_y)

List of References

[1] "Athena 111m integrated flight control system." http://www.rockwellcollins.com/sitecore/content/Data/Products/Controls/Flight_Controls/At

[2] "CGIAR-CSI SRTM 90m DEM digital elevation database." http://srtm.csi.cgiar.org/.

[3] "World geodetic system - wikipedia, the free encyclopedia." https://en.wikipedia.org/wiki/World_Geodetic_System.

[4] "pyproj - python interface to PROJ.4 library - google project hosting." http://code.google.com/p/pyproj/.