VISION BASED 3D OBSTACLE DETECTION USING A SINGLE CAMERA FOR ROBOTS/UAVs

A Thesis
Presented to
The Academic Faculty

by

Syed Irtiza Ali Shah

In Partial Fulfillment
of the Requirements for the Degree
Masters of Science in the
George W Woodruff School of Mechanical Engineering

Georgia Institute of Technology
August 2009

VISON BASED 3D OBSTACLE DETECTION USING A SINGLE CAMERA FOR ROBOTS/UAVs

Approved by:

Dr. Harvey Lipkin, Co-Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Eric N Johnson, Co-Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Nader Sadegh, Member
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: May 30th 2009

ACKNOWLEDGEMENTS

I would like to thank my committee members Dr Harvey Lipkin, Dr Eric Johnson and Dr Nader Sadegh for evaluating this thesis and for providing me requisite guidance and support, in order to successfully complete this thesis work.

I also owe a lot of thanks to my family members and friends, whose support, well-wishes and prayers brought me here.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

r	radius of flight
ω	angular velocity
Φ_c	installation angle of camera on the UAV
x,y,z	position coordinates of the camera
$\dot{x}, \dot{y}, \dot{z}$	velocity coordinates of the camera
ΔPosition	position error
ΔVelocity, ΔOrientation, ΔAngRate	velocity, orientation & angular rate errors
QPosition	process noise covariance (for position)
QVelocity, QOrientation, QAngRate	respective process noise covariances
$\phi, \theta, \psi, \dot{\phi}, \dot{\theta}, \dot{\psi}$	orientation and orientation rates of the camera
L_{bv}	rotation matrix from vehicle frame to body frame
J	error index for z-test correspondence
C	variance for error index
P	estimation error covariance matrix
Z	projected measurement vector onto image plane
x_k	projected database corner vector onto the image plane at step k
C_X, C_Z	components of variance for error index
d	residual vector in image plane
f	focal length of the camera
y_k, z_k	screen coordinates of projected database corner vector x_k at step k
X_{ck}, Y_{ck}, Z_{ck}	coordinates of position of the camera at step k
dx, dy	components of residual vector in image plane
H_k	observation matrix for extended kalman filter

R_k	covariance of the observation error
X_k^-, P_k^-	state and its error covariance for predict stage of Extended Kalman Filter
I_2	a 2x2 identity matrix
a	forward velocity
t	time in seconds
N_X	number of stored feature points in 3D space
N_Z	number of feature points observed in an image
N_I	number of images utilized

LIST OF ABBREVIATIONS

3D	three dimensional
UAV	Unmanned Air Vehicle
FOE	Focus of Expansion
EKF	Extended Kalman Filter
NED	North-East-Down

SUMMARY

Obstacle detection is a pre-requisite for collision-free motion of robots and UAVs in three dimensional (3D) space. Vision based obstacle detection and avoidance has been an active area of research in the recent past. Most research has been done for two dimensional planar motion of ground robots and using some kind of active sensors e.g. laser range finders, sonar, radar etc. Passive camera based research has mostly been done, either using stereo vision (multiple cameras) or, by developing a prior expectation map of the world and its comparison with the new image data.

In this work, an attempt has been made to find a 3D solution of the obstacle detection problem using a single camera, in an unknown world, i.e. finding size and location of the objects in the 3D world by generating a 3D scene model from the 2D information received from the camera. Once such a 3D model of the scene is obtained, an obstacle avoidance maneuver could be based on this knowledge of the size and location of obstacles. The motivation behind such an endeavor is the fact that a single camera would be carried by almost all robots or UAVs anyway, so why not use the same camera for obstacle detection and avoidance tasks as well? The attempted work would in turn facilitate building low cost, light weight miniature robots and UAVs and would obviate the requirement of additional equipment for the task.

An algorithm has been proposed in this work which uses equations of motion of the camera to track flight path, Z-test for correspondence between estimated feature points and new measurements, and Extended Kalman Filter for estimation. It finally comes up with the 3D representation of the scene. The proposed algorithm has been applied on two categories of problems.

First is the 3D detection of obstacles while following a lateral flight path around the obstacles. Simulation results show that the proposed algorithm can successfully generate a 3D model of the scene from 2D images from a single camera, in an unknown world. Further, this has been achieved by flying through a very small arc as compared to capturing full side, rear or top views, as in a typical ‘Structure from Motion’ problem.

The second case where the proposed algorithm has been applied is that of a forward flight path towards the obstacles in the scene. Accuracy of 3D information from forward motion was earlier considered to be unusable for all practical fields of view. However, the simulation results from the proposed algorithm indicate that successful 3D scene modeling is possible even with a forward flight towards the obstacle, provided that the obstacles do not lie entirely and exactly on the focus of expansion. This is the most significant contribution of this work.

CHAPTER 1

INTRODUCTION

Vision Systems in General

A vision system (as defined by Marr)[1] is a “Process that creates, given a set of images, a complete and accurate representation of the scene and its properties”. This definition is considered ‘general vision’, as the extracted representation of the scene has to be as general as possible. There are two approaches to scene representation. First is an accurate and complete representation of an observed world. This requires large amount of computational power, but gives much more information utilizable for a large range of problems, as compared to the second approach: i.e. knowing only the information specific to the problem being solved. For example, specifically for obstacle avoidance task of an indoor ground robot, the robot may only need to know which regions of its way ahead are occupied by the obstacles. Information like the shape of objects, their absolute positioning in the world, or the understanding of the relationships between these objects may not be required for this specific problem. However, if we are talking of three dimensional (3D) space of a flying robot, all such information may be relevant, in order to utilize the third dimension of altitude and to join back the planned optimal trajectory safely, past the obstacle.

Typically (Figure 1.1), a vision system includes (but is not limited to) an image acquisition mechanism, followed by an image processor and an image segmentation system. This may be followed by some kind of image reasoning, which ultimately culminates into scene modeling, comprising of desired attributes of the scene. Once the desired scene model is obtained or updated, decisions may be taken on how to achieve the vision system perceived goals.

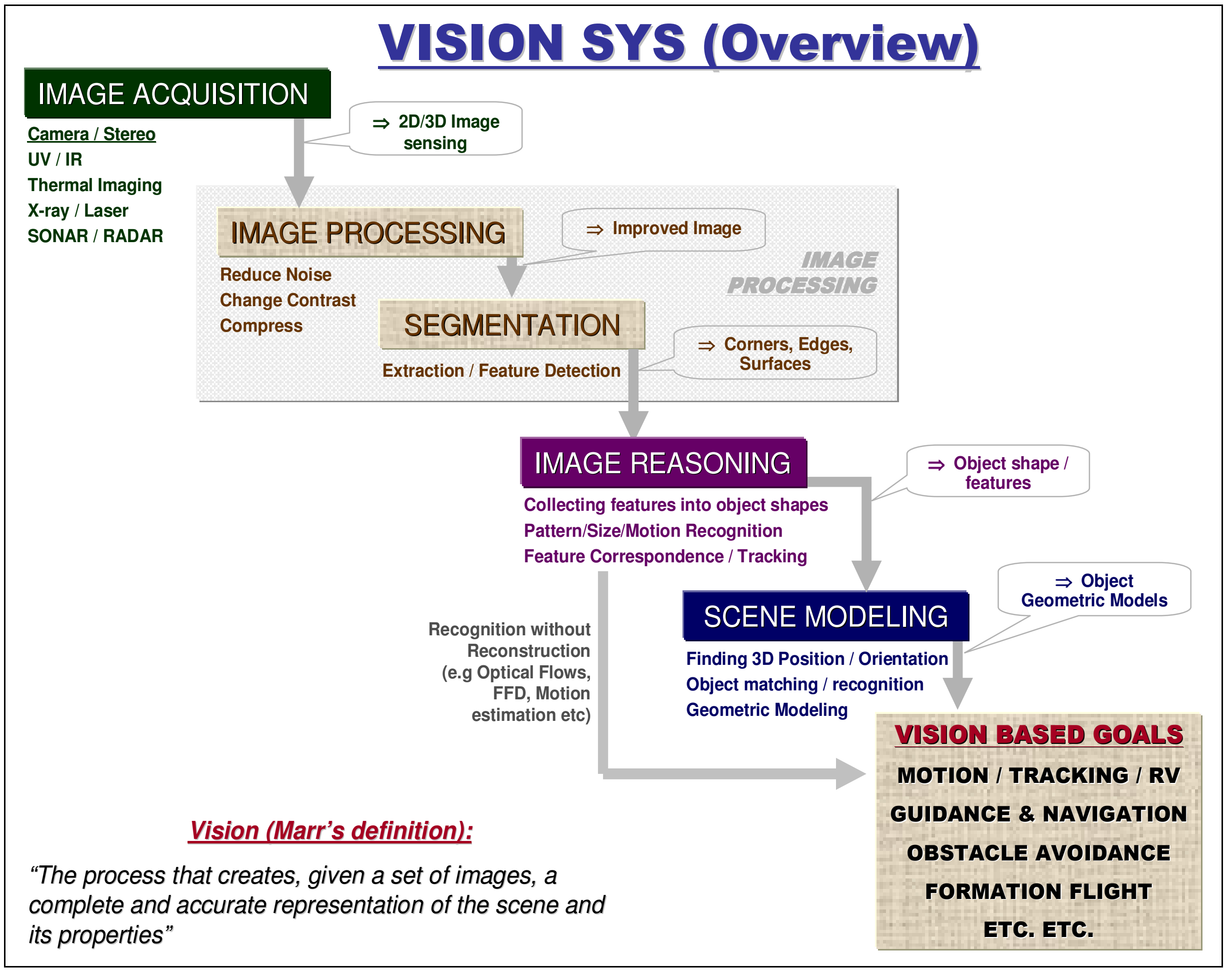

VISION SYS (Overview)
IMAGE ACQUISITION
Camera / Stereo
UV / IR
Thermal Imaging
X-ray / Laser
SONAR / RADAR
⇒ 2D/3D Image sensing
IMAGE PROCESSING
Reduce Noise
Change Contrast
Compress
⇒ Improved Image
IMAGE PROCESSING
SEGMENTATION
Extraction / Feature Detection
⇒ Corners, Edges, Surfaces
IMAGE REASONING
Collecting features into object shapes
Pattern/Size/Motion Recognition
Feature Correspondence / Tracking
⇒ Object shape / features
Recognition without Reconstruction (e.g Optical Flows, FFD, Motion estimation etc)
SCENE MODELING
Finding 3D Position / Orientation
Object matching / recognition
Geometric Modeling
⇒ Object Geometric Models
VISION BASED GOALS
MOTION / TRACKING / RV
GUIDANCE & NAVIGATION
OBSTACLE AVOIDANCE
FORMATION FLIGHT
ETC. ETC.
Vision (Marr's definition):
"The process that creates, given a set of images, a complete and accurate representation of the scene and its properties"

Figure 1.1: Overview of A Typical Computer Vision Problem

Why Vision-Based Obstacle Detection

Successful motion through 3D space requires that any objects in the flight path be avoided. This is not a prominent issue, when motion is at high altitudes. (Even birds are rarely seen above 8000ft.) On the contrary, however, if the motion is close to the ground, e.g. within 1000ft above ground level, obstacle avoidance is a very serious consideration. The various perceived roles for unmanned air vehicles fall within this flight altitude range. Such roles include (but are not limited to) Disaster Management (e.g. in fires, earthquakes, floods, landslides, volcano eruptions, storms etc) and Military or similar applications (e.g. reconnaissance, target identification, rendezvous, and nuclear, chemical, biological and conventional warfare) [26]. Such perceived roles and tasks for UAVs require various capabilities like Navigation and Control, Tracking, Terrain mapping, Formation Flying, Guidance etc, none of which is possible unless a collision free motion through the 3D space is ensured. Further, for all such roles, the robot/UAV essentially requires some kind of environment or scene sensing, which directly leads us to the requirement of vision-based systems.

In fact, vision systems are one of the most general sensors for robots and UAVs, since these deliver richer and more complete information than other sensors. For navigation in an unknown world, obstacle detection and avoidance is a fundamental behavior, which is a pre-requisite to build more complex navigational abilities. Hence the Vision-Based Obstacle detection and avoidance directly contributes to a safe operation of a robot/UAV.

Vision based obstacle detection is also motivated by nature. Most animals employ some kind of vision systems for obstacle detection and avoidance. Humans for example, use two eyes which are remarkable stereo vision sensors, giving us sufficient 3D scene information. Optical flows are employed by flying insects, which give them incredible navigational capabilities. Vision systems for ants and bees are known to gain the compass direction to important places from polarization patterns of the blue sky [2].

CHAPTER 2

PROBLEM DESCRIPTION

The overall problem of 3D Obstacle Detection may be broken down into following sub-problems.

Sensor Calibration and Image Acquisition

An image acquisition system generally comprises of one or more digital cameras. However, other sensors, aiming at specific attributes of the objects in the scene are also common. Examples include thermal imaging sensors, optical flow sensors, sonar, radar, laser range finders, infra-red sensors, ultraviolet sensors, etc. Camera or other Sensors in almost all cases require calibration (finding out intrinsic and extrinsic parameters for the sensor), before these could be effectively utilized for the problem at hand [3]. As the approach in this thesis specifically targets the GeorgiaTech GTMax UAV (figure 2.1), which already is equipped with the requisite system (hardware & software) [4], hence, this sub-problem has not been addressed in this thesis. The video images acquired on the GTMax are being digitized using a Frame Grabber.

Image Processing and Segmentation

The Image processing aims at improving images by attempting to reduce noise as far as possible, enhance contrast to a desired level, and even do data compression if required. This is followed by Image Segmentation, which extracts useful features from the output images of an Image Processor. Hence, various features points, edges, corners, surfaces, blobs, etc. are identified and located, as a result of image segmentation. For this thesis, image processing and segmentation have been taken from earlier work [4][5].

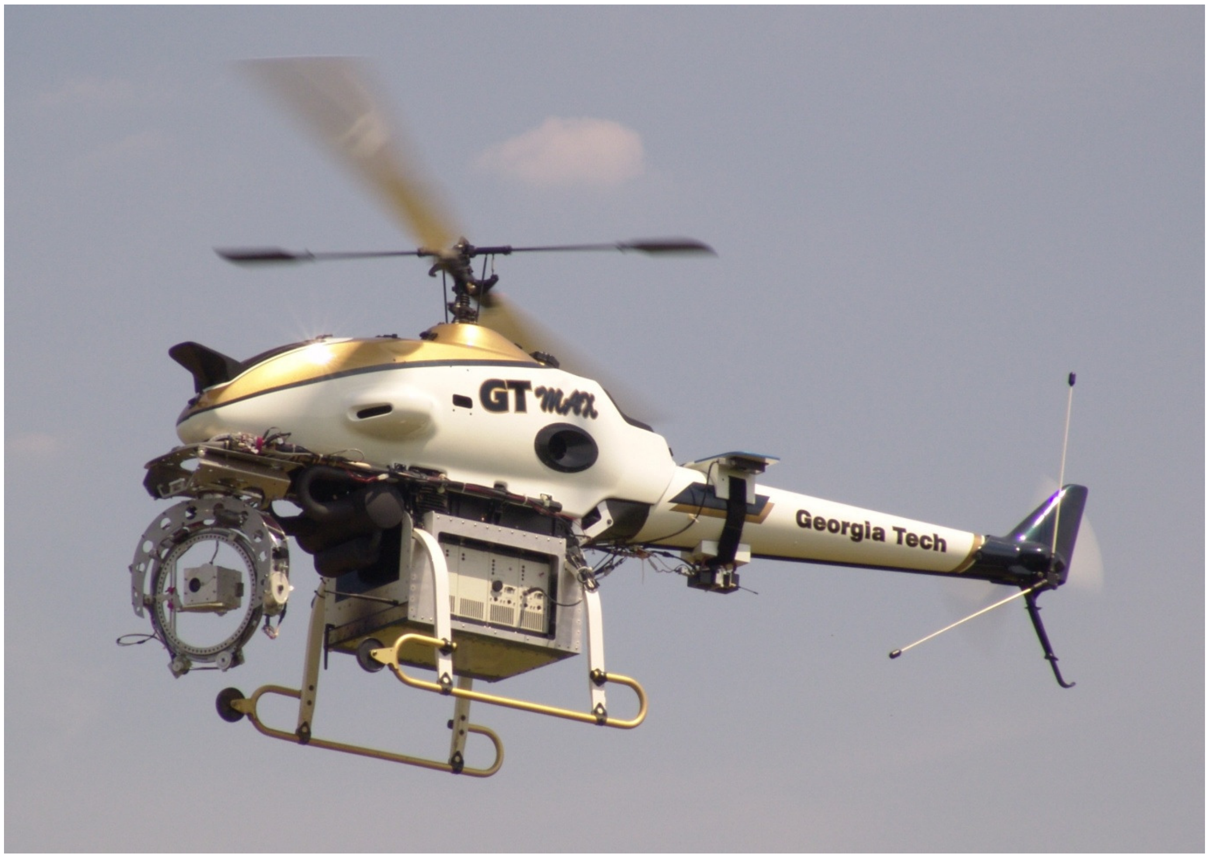


Figure 2.1 : GeorgiaTech GTMax UAV – Potential test vehicle for approach developed here

Correspondence

The next step after image segmentation is that of image reasoning, which involves collecting identified features into object shapes. Subjects of Pattern recognition, size/motion recognition, feature tracking and feature correspondence, all can be viewed as various forms of Image reasoning. Z-test [6] has been used in this work, to solve the problem of feature correspondence as was proposed in Ref [7]. Z-test is a statistical test used in inference to determine if the difference between a sample mean and the population mean is large enough to be statistically significant, that is, if it is unlikely to have occurred. In order for Z-test to be reliable, certain conditions must be met, most important of which, is that the population standard deviation must be known. Further, the sample must be a random sample, with a normal distribution of population sampling. In actuality, knowing the true standard deviation of a population is unrealistic (in which case a t-test must be used). However, in the case here, as the entire population of segmented feature points is known exactly, Z-test is the preferred choice.

Noise and Non-Linearities

Noise in the image data is generally modeled as some random variations in brightness information. Such noise can originate in film grain, or in electronic noise in the input device (digital camera or other image acquisition media) sensor and circuitry, or in the unavoidable shot noise of an ideal photon detector[8]. For all the simulations in this thesis, random Gaussian noise has been added so as to bring the simulations close to real scenarios. Further, due to the non-linear system equations (see Chapter 4), Extended Kalman Filter has been chosen, which can treat this noise explicitly.

3D versus 2D

Solving a two dimensional problem of obstacle detection and avoidance, (which is the case for most ground robots) is relatively simpler than solving a 3D problem. The 2D problem deals with the intensity map at each pixel on an image, in which, obstacles are identified that need to be avoided. Subsequent images indicate changes in the scene, which update this information for obstacle avoidance. The 2D obstacle detection hence generally solves only the problem of 'directions to avoid' and need not generate a scene model. For the specific case of obstacle avoidance, the robot may only need to know the regions of its way ahead that are occupied by obstacles. No information like shape of the objects, their absolute positioning in the world or the understanding of the relationships between these objects is required. Consequently the image data may be directly used without a reconstruction of the three-dimensional world of motion. Therefore, no explicit knowledge about the camera parameters, ego-motion, and camera-to-ground coordinate transformations is required [9]. On the contrary, a general 3D obstacle detection problem solves for all such attributes of world, and this is what has been addressed in this thesis.

Avoidance Maneuver in 3D

Once the problem of locating feature points in the 3D world is solved, we get the 3D coordinate information for all identified object features in the scene, from which we generate a 3D model of the environment of the UAV. This is obviously far more computationally expensive, than a 2D case, but still is a preferred choice in this thesis, due to the detailed information of the environment, we obtain.

The knowledge of the 3D environment then enables an Obstacle avoidance maneuver to be generated. This involves leaving the previous trajectory to avoid the unexpected obstacle, and then joining back the original trajectory in 3D space when past the obstacle, with minimum effort. This problem has been addressed in references [10] & [24].

CHAPTER 3

APPROACHES TO OBSTACLE DETECTION

There has been an extensive literature addressing obstacle detection and avoidance, particularly for ground robots. Various approaches to obstacle detection are roughly categorized here, under the following headings (refer Figure 3.1 also).

Multiple Sensor Based Obstacle Detection & Avoidance

The most common approach to obstacle detection & avoidance is that of use of multiple sensors. Thus for example, David Coombs and Karen Roberts [11] propose two cameras looking obliquely to steer between objects. The left and right proximities have been compared to steer through the gap.

Another similar development is a vision system capable of guiding a robot through corridor-like environments by Argyros and Bergholm [12]. It uses three cameras, one for central forward vision and the other two for peripheral vision. The main principle is to implement a honey-bee-like reactive centering behavior by controlling the movement in a way that the optical flow on both sideward-looking cameras is equal. The normal flow for all three cameras is computed by an intensity-based algorithm, after which, the depth to obstacles visible in the periphery cameras is extracted by using the central camera to compensate for the rotational component of the ego-motion. It may be seen that the hardware requirements for this approach are that of three cameras and two workstations in order to compute the three optical flows.

Analogous approaches have been proposed and successfully applied for various robotic platforms. Representative examples are Ref [13] for Stereo Vision (most common for ground robots) and Ref [14] for fusing Radar and Vision for obstacle avoidance on cars.

Single Sensor Based Approaches

In his PhD thesis [18] and relevant published work [19] [20], Randal C Nelson proposes the use of certain measures of flow field divergence as a qualitative cue for obstacle avoidance. It has been shown that directional divergence of the 2D motion field indicates the presence of obstacles in the visual field of an observer, undergoing generalized rotational and translational motion. Divergence information has been calculated from image sequences, based on the directional separation of optical flow components and the temporal accumulation of information. The use of the system to navigate between obstacles has been demonstrated by experimental results. This approach essentially does not do obstacle detection in 3D space, but instead comes up with a 'No-Go' direction, skipping directly to the obstacle avoidance part.

In their paper [21], Young et. Al. present an approach to obstacle detection, using optical flow without recovering range information. A linear relationship, plotted as a line called reference flow line, has been used to detect discrete obstacles above or below the reference terrain. The parameters of the reference flow line are estimated using the optical flow of a specific part of the picture that is assumed to be obstacle-free. Slopes of surface regions have also been computed. Objects that intersect with the reference space line and occlude it cause different flow values than the reference line and can thus be detected. It may be seen that this approach may work effectively for ground robots in general, and for UAVs during landing, but does not seem very useful in normal 3D flight of a robotic UAV, primarily because of absence of any reference or obstacle free terrain data in completely unknown flying environments.

Nicholas Hatsopoulos and James Anderson [22] also use optical flow, but instead calculate time to contact, which is an optical property. However, they themselves describe in the paper that this approach, which has been proposed for collision avoidance in cars, is not effective in realistic driving environments, when the surfaces are not very flat and are not perpendicular to the center of camera axis.

Three Dimensional Approaches

Nakao et al [23] present a method of 3D shape reconstruction of objects for a camera mounted on a robotic arm and the object being modeled on the turn table. This paper does not seem to address the correspondence problem in detail, (probably because there are very few feature points in the scene in such structured environment). Further, this approach effectively uses a single camera and an Extended Kalman Filter for 3D shape reconstruction. Besides, there had been a lot of literature under the heading of 'Structure from Motion' problem. The problem at hand may be considered as one case of such a problem.

Obstacle Detection & Avoidance in Structured Environment

Ilic et al [15], present a monocular ground plane obstacle detection method using optical flow anomalies. The optical flow is computed on a single image row and compared to a model for ground point optical flow, obtained by direct calibration. This approach seems efficient for ground robots but may not be suitable for UAVs, as the model for ground point optical flow may not be obtained for a completely unknown / unstructured 3D environment, which a UAV is expected to fly into. Ref [16] and [17] also present approaches for obstacle detection and avoidance in either structured or partially known environments.

Proposed Approach

The problem attempted in this thesis is that of a single sensor, which is a camera and the solution being sought here in this thesis is that for a 3 dimensional problem in perfectly unknown world. The details of the proposed approach are described in the following chapter.

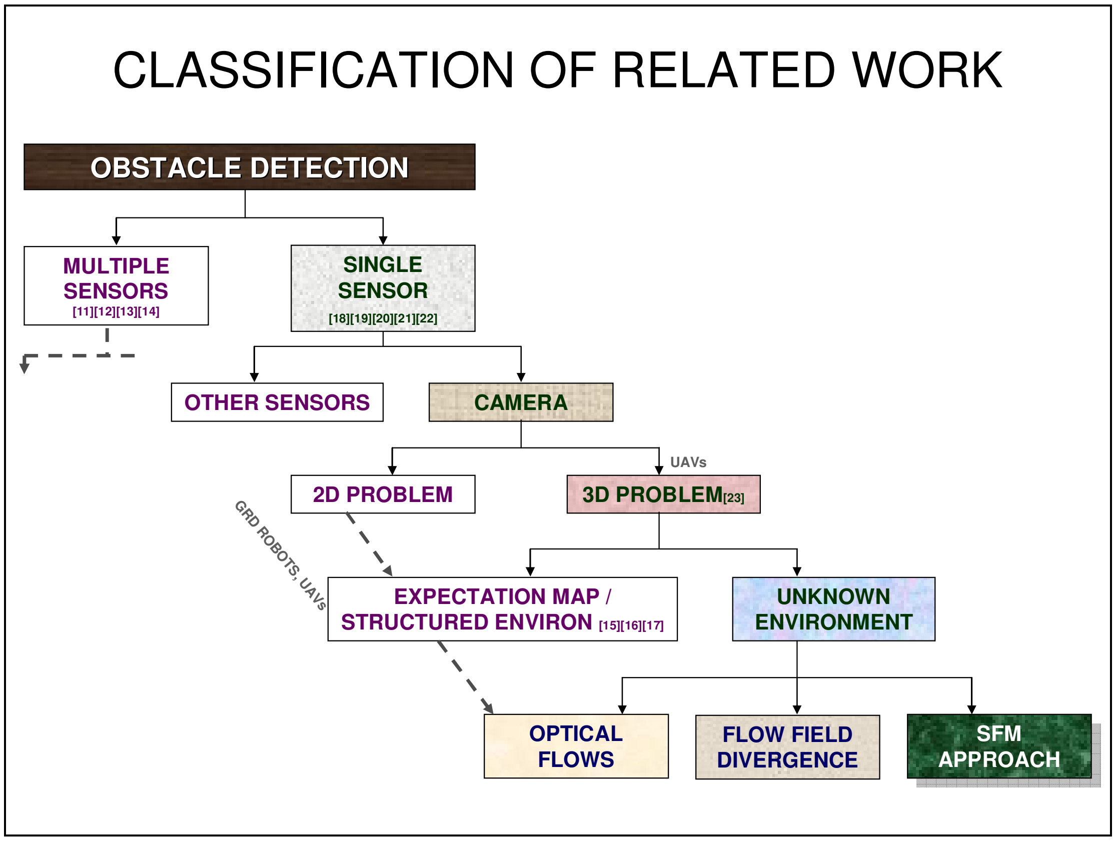

CLASSIFICATION OF RELATED WORK
OBSTACLE DETECTION
MULTIPLE SENSORS [11][12][13][14]
SINGLE SENSOR [18][19][20][21][22]
OTHER SENSORS
CAMERA
UAVs
2D PROBLEM
3D PROBLEM[23]
GRD ROBOTS, UAVs
EXPECTATION MAP / STRUCTURED ENVIRON [15][16][17]
UNKNOWN ENVIRONMENT
OPTICAL FLOWS
FLOW FIELD DIVERGENCE
SFM APPROACH

Figure 3.1 : Classification of Related Work

CHAPTER 4

PROPOSED 3D OBSTACLE DETECTION: LATERAL FLIGHT

Equations of Camera Motion

For the present problem, it is supposed that a camera is capturing 2D images and is mounted on a UAV. Immediately after the detection of feature points in the scene, UAV stops its forward flight and instead starts flying around the object, following a circular path, where the flight path is tangent to the radial vector to the object. UAV flies in a radius of flight '*r*', with angular velocity 'ω' at a constant altitude 'h'. The relative position of the camera in 3D space is 'x', 'y', 'z' and its orientation is 'φ', 'θ' and 'ψ' (Refer Figure 4.1 below). This is an extreme case of obstacle avoidance maneuver selected to maximize predicted ability to generate the 3D map.

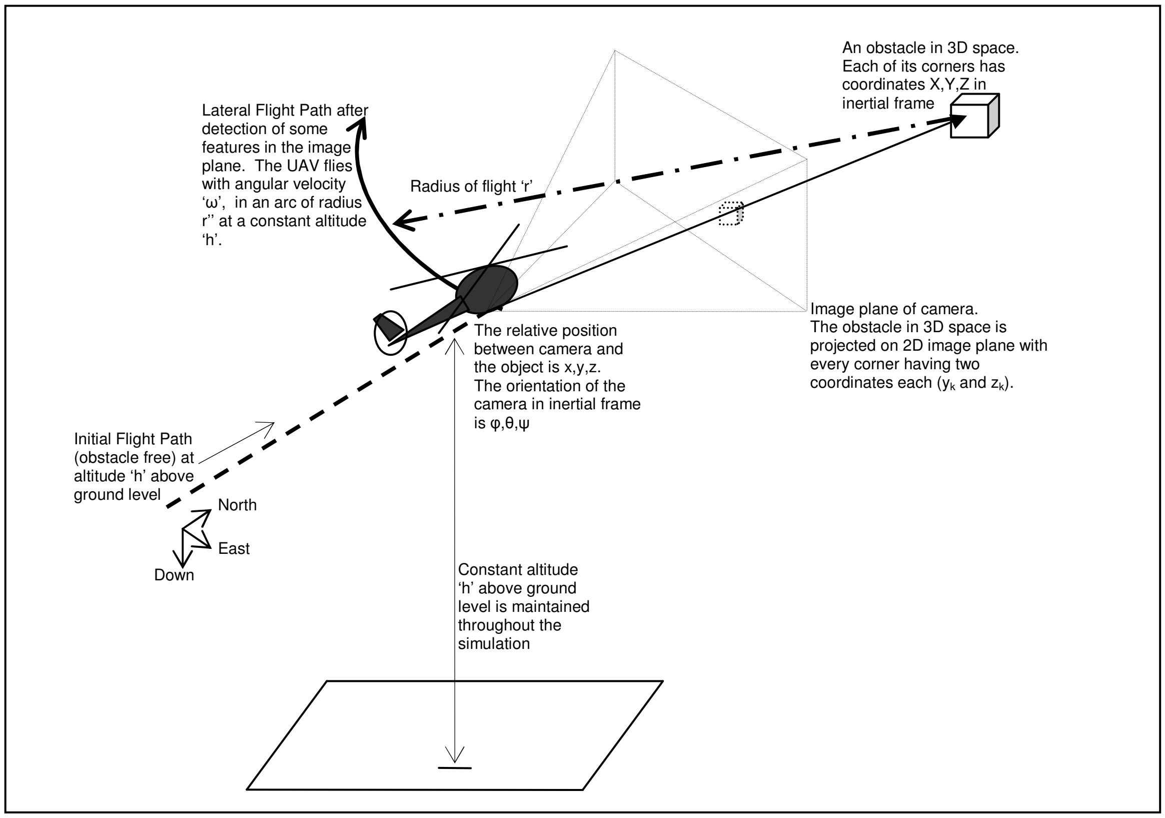

An obstacle in 3D space. Each of its corners has coordinates X,Y,Z in inertial frame
Lateral Flight Path after detection of some features in the image plane. The UAV flies with angular velocity 'ω', in an arc of radius r'' at a constant altitude 'h'.
Radius of flight 'r'
Image plane of camera. The obstacle in 3D space is projected on 2D image plane with every corner having two coordinates each (y_k and z_k).
The relative position between camera and the object is x,y,z. The orientation of the camera in inertial frame is φ,θ,ψ
Initial Flight Path (obstacle free) at altitude 'h' above ground level
North
East
Down
Constant altitude 'h' above ground level is maintained throughout the simulation

Figure 4.1: Camera Mounted on a UAV with a Detected Object in the Scene

With the vehicle frame of reference as North-East-Down (NED), the following states and their rates are obtained for the camera

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r\sin\omega t \\ r\cos\omega t \\ -h \end{bmatrix} + \Delta Position$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} r\omega\cos\omega t \\ -r\omega\sin\omega t \\ 0 \end{bmatrix} + \Delta Velocity$$

(4.1)

where *x,y,z* are the position states, with dot notation specifying the rate and *ΔPosition* and *ΔVelocity* are the error values for position and velocity vectors modeled as Gaussian noise vector of size 3x1, respectively. (Values of the noise covariances have been chosen keeping in view similar calculations e.g. in Ref [24]).

The orientation and orientation rates of the camera are given by

$$\begin{bmatrix} \varphi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \varphi_c \\ \theta_c \\ \psi \end{bmatrix} + \Delta Orientation$$

$$\begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\omega \end{bmatrix} + \Delta AngRate$$

(4.2)

where φ, θ, ψ define the orientation of the camera on the UAV, φ_c is the installation angle of camera on UAV, dot notation specifies the rate and *ΔOrientation* & *ΔAngVelocity* are the noise values for Orientation and orientation rate modeled as Gaussian noise vectors of size 3x1, respectively.

For conversion between body frame and vehicle frame, the rotation matrix is as follows

$$L_{bv} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (4.3)$$

and

$$L_{vb} = L_{bv}^{T} \tag{4.4}$$

Z-test for Correspondence

Statistical Z-test method has been used to solve the correspondence problem between the estimated corners from database and the measurements. The Z-test has been taken for a certain error index (J) and is the square of this index divided by its variance (C) i.e $Zvalue=J^2/C$. Both the estimation error covariance (matrix P) and the measurement error covariance (matrix R) have been taken into account while calculating C. Then the Z-test value is inversely related to the likelihood of an event that a given measurement corresponds to the corner point chosen. Thus for example, if there is a large error between the measurement and the image data, but the measurement also has a large uncertainty, then the probability of its correspondence should be higher than the case in which, the measurement has a small uncertainty. Thus each corner point is to be assigned to a point, which attains the least Z-test value, meaning thereby, the highest likelihood.

In the Figure 4.1, Z is the projected measurement vector onto image plane and x_k is the projected database corner vector onto the image plane. Hence

$$J = dx^2 + dy^2$$

$$C = C_X P C_X^{T} + C_Z R C_Z^{T} \tag{4.5}$$

where

$$C_X = \frac{\partial J}{\partial X_v} \quad \text{and} \quad C_Z = \frac{\partial J}{\partial Z} \tag{4.6}$$

are the two components of the variance C of the error index J

It may be noted from Fig 4.2, that the residual vector is

$$d = Z - x_k \tag{4.7}$$

Pin-Hole Camera Model

Assuming that the camera is mounted at the center of gravity of the vehicle, let L_{bv} denote a known camera attitude represented by a rotation matrix from inertial to the camera frame. A camera frame is taken so that the camera's optical axis aligns with its X_c axis. Then the relative position in camera frame will be

$$\boldsymbol{X} = \boldsymbol{X_v} - \boldsymbol{X_p} \quad \text{(in inertial frame)} \tag{4.8}$$

$$\boldsymbol{X_c} = L_{bv}\,\boldsymbol{X} \quad \text{(in camera frame)} \tag{4.9}$$

where

$$\boldsymbol{X_c} = [X_c(t) \quad Y_c(t) \quad Z_c(t)]^T \tag{4.10}$$

and the subscript 't' is used for the target or the object to be modeled, subscript 'c' is used for the camera and upper case bold $\boldsymbol{X}$ indicates a 3x1 vector.

Assuming a pin-hole camera model as shown in the Figure 4.2, the object position in the image at a time step t_k is given by (x_k is a 2x1 vector)

$$x_k = \begin{bmatrix} y_k \\ z_k \end{bmatrix} = \frac{f}{X_{ck}} \begin{bmatrix} Y_{ck} \\ Z_{ck} \end{bmatrix} \tag{4.11}$$

This equation is non-linear with respect to the relative state. (Hence an Extended Kalman Filter has been used here).

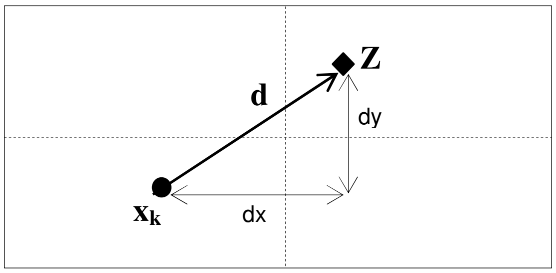


Figure 4.2: The Residual Vector

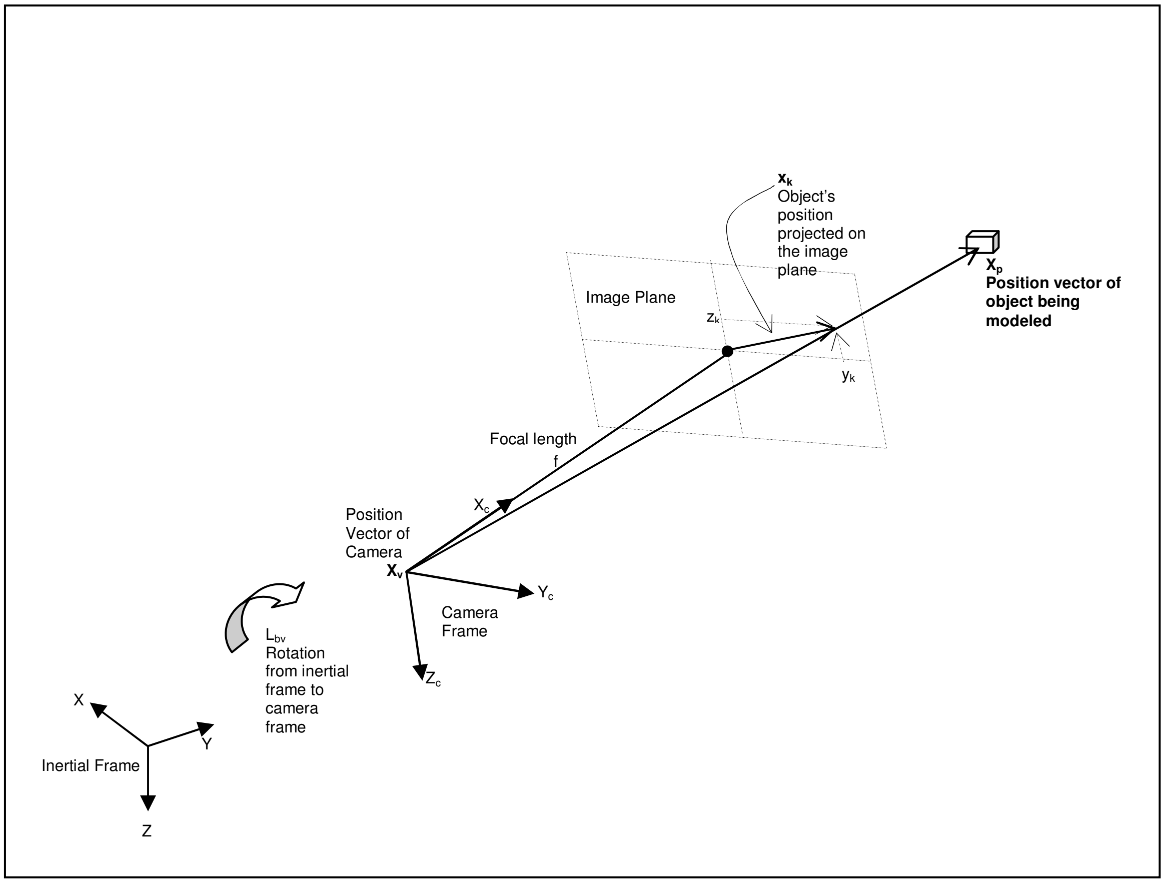

x_k
Object's position projected on the image plane
X_p
Position vector of object being modeled
Image Plane
z_k
y_k
Focal length
f
X_c
Position Vector of Camera
X_v
Y_c
Camera Frame
Z_c
L_bv
Rotation from inertial frame to camera frame
X
Y
Inertial Frame
Z

Figure 4.3: A Standard Pin-Hole Camera Model [24]

In the implementation, focal length *'f'* of the camera has been assumed to be unity without loss of generality.

Extended Kalman Filter

Following Eqns 4.5, 4.6 and 4.7, we first write expressions for the components of Variance matrix C as C_x and C_z, using chain rule as follows

$$C_X = \frac{\partial J}{\partial X_v} = \frac{\partial J}{\partial d}.\frac{\partial d}{\partial x_k}.\frac{\partial x_k}{\partial X_v} \quad \text{and} \quad C_Z = \frac{\partial J}{\partial Z} = \frac{\partial J}{\partial d}.\frac{\partial d}{\partial Z} \qquad (4.12)$$

The 'predict' and 'update' stages of Extended Kalman Filter are given as follows.

Predict Stage (before new measurement)

For an Extended Kalman Filter, the predicted state is defined by

$$X_k^- = f(X^-_{k-1}, U_k)$$

which, for this case of no dynamics and no input for the feature point being modeled, (in the predict stage before the new measurement), simplifies to

$$X_k^- = X^-_{k-1} \qquad (4.13)$$

The estimation covariance matrix for predict stage (before measurement) is defined by

$$P_k^- = F_k P^-_{k-1} F_k^T + Q_k$$

which for predict stage of no dynamics case, simplifies to

$$P_k^- = P^-_{k-1} + Q_k \qquad (4.14)$$

(since $F_k = \dfrac{\partial f}{\partial X} = I$ here, for no dynamics case)

Update Stage (after new measurement)

$$K_k = P^-_k H_k^T (H_k P^-_k H_k^T + R_k)^T \qquad (4.15)$$

$$X_k = X^-_k + \sum K_k d \qquad (4.16)$$

$$P_k = P_k^- - \sum K_k H_k P_k^- \qquad (4.17)$$

where R_k is the measurement error covariance matrix and the observation matrix Jacobian H_k , which is defined as partial derivative of the residual (***d***) with respect to partial derivative of the state (***X***) is calculated here as

$$H_k = \frac{\partial d}{\partial X} = \frac{\partial d}{\partial x_k} . \frac{\partial x_k}{\partial X} \qquad (4.18)$$

For the vectors ***d***, ***X*** and $\boldsymbol{x_k}$ defined above, this is evaluated as follows (for detailed derivation of this result, please see Appendix A)

$$\frac{\partial d}{\partial x_k} = -I_2 \qquad \text{and} \qquad \frac{\partial x_k}{\partial X} = \frac{1}{x_{ck}} \begin{bmatrix} -x_k & I_2 \end{bmatrix} \qquad (4.19)$$

where I_2 denotes a 2x2 identity matrix.

Thus H_k turns out to be

$$H_k = -\frac{1}{X_{ck}} I_2 \left[-x_k \quad I_2\right] \tag{4.20}$$

Similarly, the covariance matrix R_k (measurement error covariance) is defined as

$$R_k = \frac{\partial d}{\partial Z} R \frac{\partial d^T}{\partial Z} \tag{4.21}$$

which is evaluated as (for detailed derivation of this result please see Appendix A)

$$R_k = I_2 R I_2 \tag{4.22}$$

where R is the measurement noise covariance.

It may be seen now that the components of Variance matrix C, as C_x and C_z given above, may be evaluated as (for details, see Appendix A)

$$C_X = \frac{\partial J}{\partial X_v} = \frac{\partial J}{\partial d} H_k = 2d^T H_k \tag{4.23}$$

and

$$C_Z = \frac{\partial J}{\partial Z} = \frac{\partial J}{\partial d}\frac{\partial d}{\partial Z} = 2d^T I_2 \tag{4.24}$$

3D Modeling Algorithm

The above equations have been implemented in the 3D scene modeling algorithm as shown in Fig 4.3. The algorithm starts when a feature point is detected in the scene. This information is being fed to the program by the frame grabber after image processing and segmentation. Further, the camera calibration information is also being fed to the program by GTMax onboard systems, which includes its location in 3D space and installation angle on the UAV, besides the knowledge of its FOV (Field of View) and its image plane size. The UAV then starts flying in a circular path in radius *'r'*. Equations 4.1 – 4.4 give the position, orientation and respective rates for the camera and equations 4.8, 4.9 and 4.10 position information in camera and inertial frame. For the first iteration, as the estimation database is empty, all the feature points as measured in the image frame, go into the estimation database without establishing any correspondence. Since this was only 2D information from the image plane, the third dimension is unknown and is supposed to be zero i.e. all points are supposed to lie on the ground plane initially. When the subsequent image information is received, the estimation points in the database are projected onto image plane (via equation 4.11) and the residual vector is calculated between the new measurement and the estimated points on image plane (equation 4.7). Z-test correspondence is done to establish which measurement corresponds to which estimated value and the new values are updated with the extended Kalman filter (equations 4.13 to 4.24). If correspondence is not established between a measured feature points in the image, with any of the estimated feature points, this feature point is recognized as a new point. Conversely if an estimated feature point existed, for which there was no corresponding measurement in the new image, this point is marked for deletion. However, it is actually not deleted unless it remains without correspondence for next consecutive N images. This has been done to ensure, that if a feature point temporarily goes out of view, it is not deleted immediately, otherwise the whole simulation time would increase, if it came back into the view later on and was instead recognized as a new feature requiring new estimation starting from ground plane.

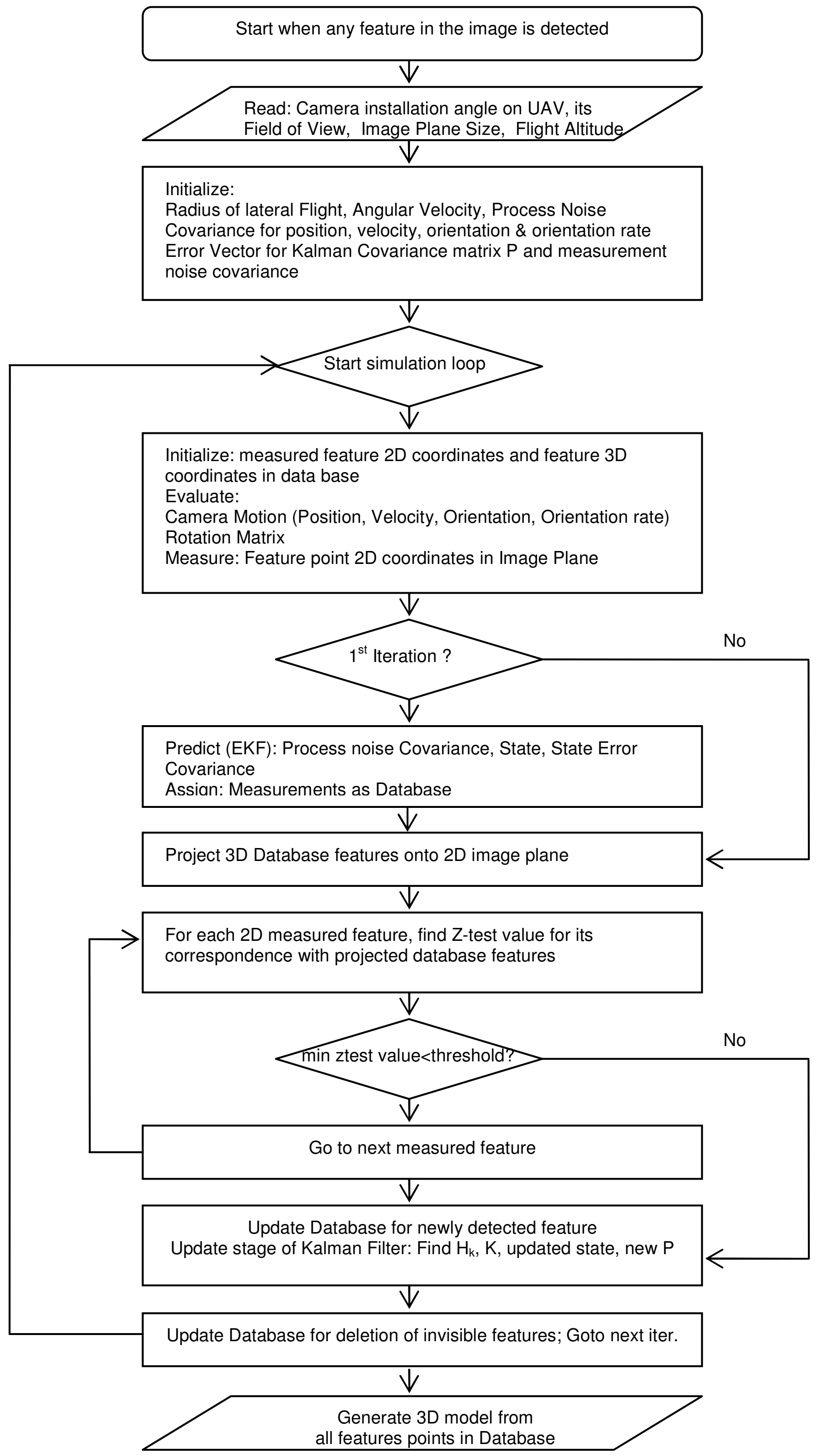

Start when any feature in the image is detected
Read: Camera installation angle on UAV, its Field of View, Image Plane Size, Flight Altitude
Initialize: Radius of lateral Flight, Angular Velocity, Process Noise Covariance for position, velocity, orientation & orientation rate Error Vector for Kalman Covariance matrix P and measurement noise covariance
Start simulation loop
Initialize: measured feature 2D coordinates and feature 3D coordinates in data base Evaluate: Camera Motion (Position, Velocity, Orientation, Orientation rate) Rotation Matrix Measure: Feature point 2D coordinates in Image Plane
1st Iteration ?
No
Predict (EKF): Process noise Covariance, State, State Error Covariance Assign: Measurements as Database
Project 3D Database features onto 2D image plane
For each 2D measured feature, find Z-test value for its correspondence with projected database features
min ztest value<threshold?
No
Go to next measured feature
Update Database for newly detected feature Update stage of Kalman Filter: Find H_k, K, updated state, new P
Update Database for deletion of invisible features; Goto next iter.
Generate 3D model from all features points in Database

Figure 4.4: The Proposed Algorithm

Simulation Results (Lateral Flight)

As a first case, a cube was selected with eight corners (or eight feature points). This known model of the cube was used to verify the ability of the algorithm to successfully generate a 3D model of this cube using the 2D image information captured from a single camera. The simulation results are as presented in fig 4.4. In this figure, the solid (magenta) lines indicate the object to be modeled, the blue diamonds indicate the progressive outcome of corner estimation from the proposed algorithm, whereas the wavy black arcs indicate the flight path of the camera. The final figure (at 60 sec) shows that the blue diamonds approach the actual corners of the object being modeled, indicating a successful 3D obstacle detection for this case.

As a next case, a scene comprising of 35 feature points was chosen, as various corners of high-risers in a typical urban scenario. The simulation results for lateral flight path are shown in Fig 4.5. In this figure also, the solid (magenta) lines indicate the object to be modeled, the blue diamonds indicate the progressive outcome of corner estimation from the proposed algorithm, whereas the wavy black arcs indicate the flight path of the camera. The final figure (at 100 sec) shows that the blue diamonds approach the actual corners of the object being modeled, indicating a successful 3D obstacle detection for this case as well.

The table on the next page gives the values used for the simulation:

Table 4.1: Values used for simulation : Lateral Flight

Flight altitude above ground level	140 ft
Radius of flight about the object	140 ft
Angular velocity around the object	0.36 deg/sec
Camera field of view	30 deg
Position error in all three states each	1%
Velocity error in all three states each	1%
Orientation error in all three states each	0.01%
Angular velocity error in all three states each	0.01%

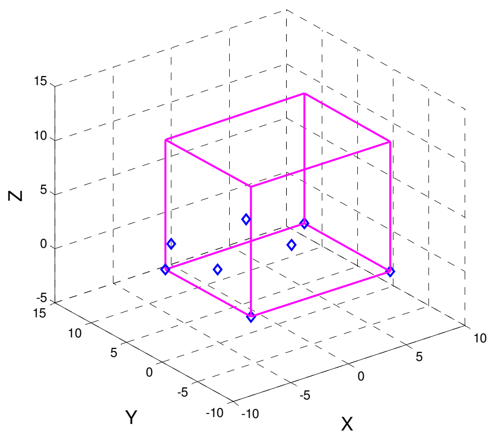
The simulation results show that

- The proposed algorithm can successfully generate a 3D model of the scene, from 2D image information.
- This modeling only requires one camera as the sensor.
- The results have been achieved for an unknown world and no constraints were put on the environment being modeled. No attributes of the environments were provided to the system, except for the 2D images being captured by the camera.

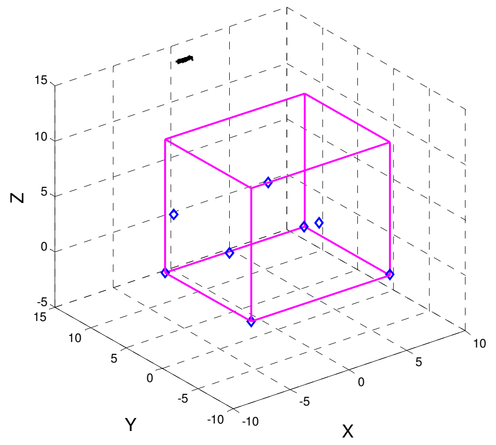
- The scene modeling has been achieved (to within $\pm 3\%$ of actual 3D locations of the feature points) in 60 seconds of flight for 8 feature points, and 100 seconds of flight for 35 feature points.

- The successful 3D scene modeling required flying through a very small arc of lateral flight as compared to the size of object being modeled. There had been no need to capture images from all sides of the objects being modeled. Thus the approach is more practical than a typical 'Structure from Motion' problem, which requires right, left, top or other views of the object, in order to generate its 3D model.

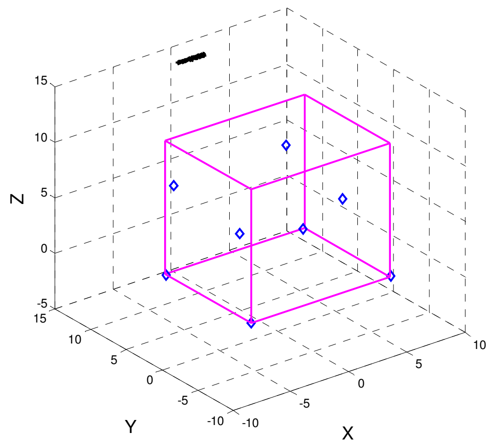
- The algorithm does require some feature points in the scene. Hence if no feature points are detected in the scene, the algorithm implies that there are no obstacles to be avoided and the initial flight path of the UAV may be continued without any disruption. This is almost always true in real world scenarios. However, there could be one exception of that of a flat wall in front, which is discussed as 'Future Work' in Chapter 7.



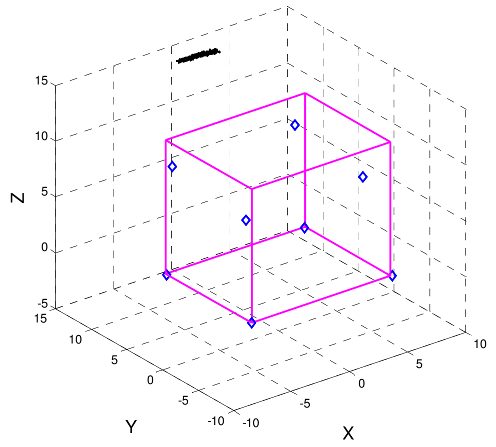
a) At time 0 sec



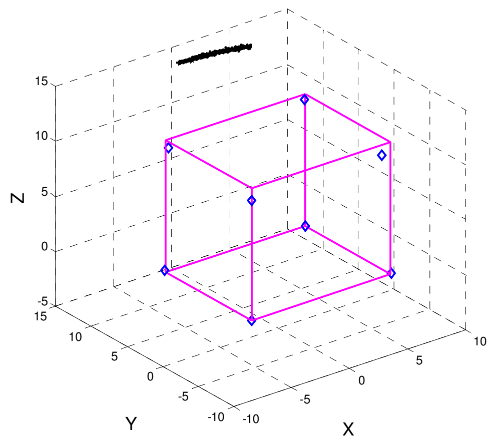
b) At time 12 sec



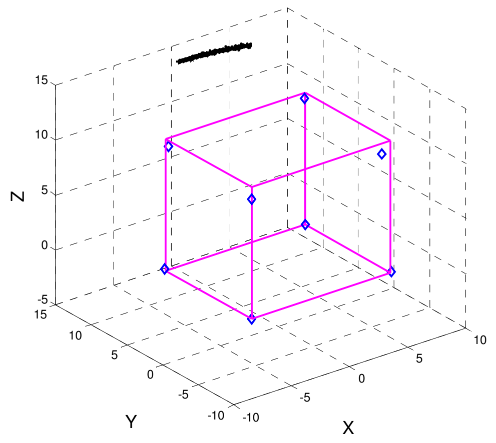
c) At time 24 sec



d) At time 36 sec

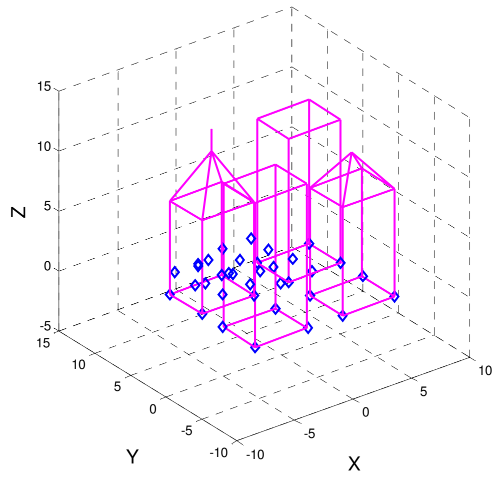


e) At time 48 sec

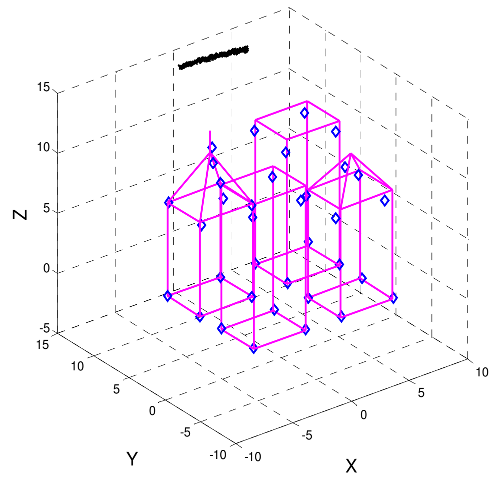


f) At time 60 sec

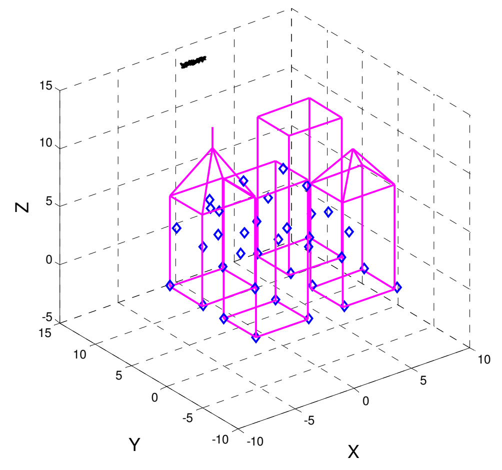
Figure 4.5: Lateral Flight Simulation Results with 8 Feature Points. Image processing is updated at 10 frames / sec. Convergence is good at 60 sec, traveling 25 deg around the object



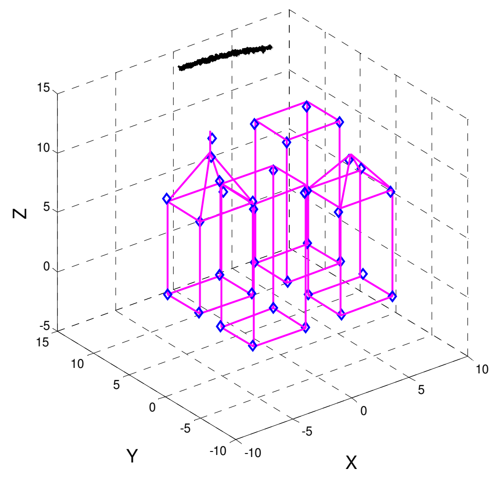
a) At time: 0sec



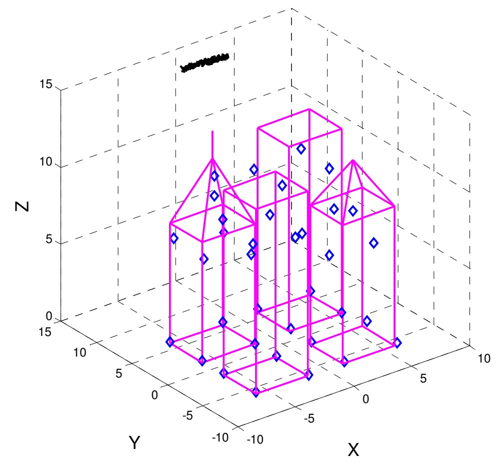
d) At time: 60sec



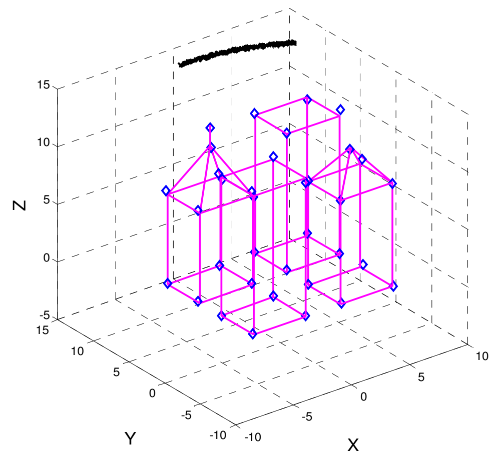
b) At time: 20 sec



e) At Time: 80 sec



c) At time: 40sec



f) At time: 100sec

Figure 4.6: Lateral Flight Simulation results with 35 feature points. Image processing is updated at 10 frames / sec. Convergence is good at 100 sec, traveling 40 deg around the object

CHAPTER 5

3D OBSTACLE DETECTION : FORWARD FLIGHT

Introduction

In their research paper[25], Matthies, Kanade and Szeliski present Kalman Filter-based Algorithms for Estimated Depth from Image Sequences. Besides other conclusions, they have shown that

1. For a translating camera, the accuracy of depth estimates increases with increasing distance of image features from the focus of expansion (FOE, which is a point in the image where translation vector pierces the image).
2. Best translations are parallel to the image plane and the worst are forward along the camera axis.
3. For practical fields of view, the accuracy of depth extracted from forward motion will be effectively unusable for a large part of the image. Thus for practical depth estimation, forward motion is effectively unusable compared with lateral motion.

Proposed Approach

Chapter 4 of this thesis demonstrated that Lateral flight gives good 3D scene modeling of objects from 2D image data. (In fact depth is just one coordinate of any feature point in 3D space). This substantiates deduction 2 above. This however, is apparently an awkward flight maneuver form a practical perspective in the sense that a UAV, which was supposed to fly forward, has to start flying laterally, as soon as some object is detected in the scene, in order to estimate its depth or 3D location in space. Hence, here an attempt was made to do depth estimation while flying forward, which is in conflict to what was recommended in conclusion 3 above. However, two facts are important here.

Firstly, estimation of 3D positions of those objects is attempted, which do not lie exactly at focus of expansion, because if the features exactly lie at FOE, there is no solution to the problem. This is in accordance with the first deduction mentioned above. We propose that if the features are not at FOE, even flying forward could give reasonable depth estimation. Of course the accuracy would improve with increasing distance of features from FOE, as stated in Ref [25] above.

Secondly, it may be noted that the conclusions in the above-referred paper were arrived at by linearizing the system equations and using a Kalman Filter. In this thesis however, it is investigated, whether the use of non-linear Extended Kalman Filter instead of a regular Kalman Filter, can provide good results for forward flight of a UAV.

Accordingly it is proposed in this work that, subject to the two considerations just mentioned above, flying forward will give depth estimation, which is of practical use, as opposed to deduction 3 of above referred paper.

Implementation of this 3D obstacle detection in forward flight, changes only the equations of motion of camera i.e. equations 4.1 and 4.2 of Chapter 4 above. All other equations presented for Lateral flight in Chapter 4 above, remain valid in this forward flight case as well. This also applies to fig 4.1 (Residual Vector), fig 4.2 (Pin-hole Camera Model) and fig 4.3 (Proposed Algorithm). The changes required in equations (4.1) and (4.2) for this case are: $\omega=0$ (for no lateral flight), 2nd (forward) component of position vector added with a factor '$a \times t$', where 'a' is forward velocity and t is time. Also second component of velocity vector is added with this constant factor 'a' (forward velocity).

To avoid the obstacles in FOE, the speed of flight is a critical factor. If it is too high, the images of the objects, which are enlarging, in this case as the motion is towards them, will quickly occupy almost whole of the image, including FOE as well. Hence the 3D scene modeling would not be possible. On the contrary if the speed of flight is too low, there is less variation in the subsequent images, and hence less new information in those images. This will in turn prolong the simulation time to an unacceptable extent. In this case of 35 feature points, the optimum speed of flight was found by iterations in repeated simulations, so as to achieve the correct 3D modeling at a relatively high speed.

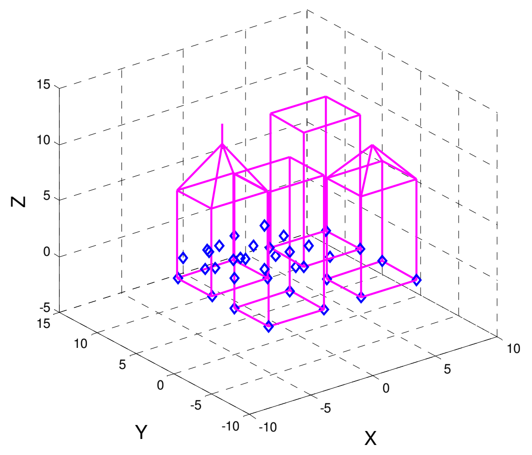
Forward Flight Simulation Results

The simulation results for 3D obstacle detection in forward flight are presented in figure 5.1. In this figure, the solid (magenta) lines indicate the object to be modeled, the blue diamonds indicate the progressive outcome of corner estimation from the proposed algorithm, whereas the wavy black line indicates the forward flight path of the camera. The final figure (at 125 sec) shows that the blue diamonds approach the actual corners of the object being modeled, indicating a successful 3D obstacle detection for this case.

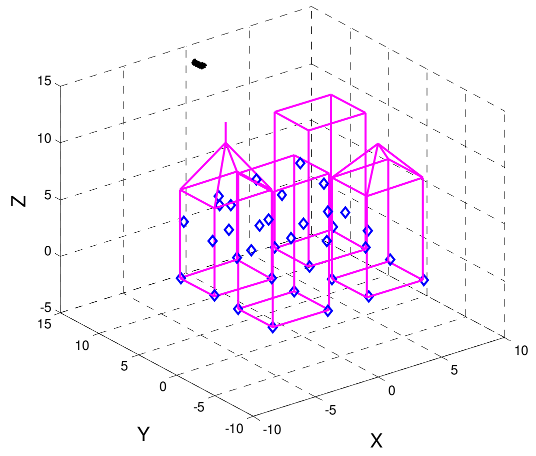
The table below gives the values used for the simulation:

Table 5.1: Values used for simulation : Forward Flight

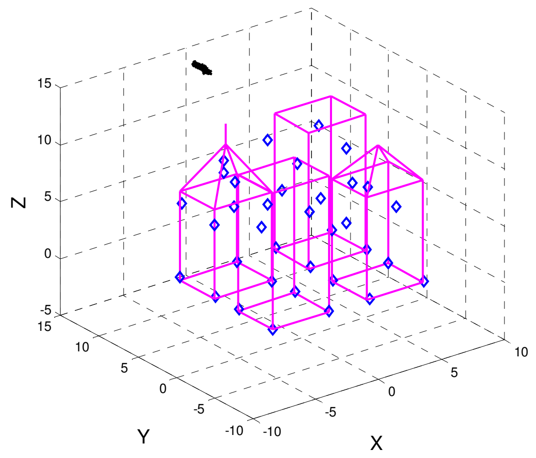
Flight altitude above ground level	140 ft
Forward Velocity	1.4 ft/sec
Camera field of view	30 deg
Position error in all three states each	1%
Velocity error in all three states each	1%
Orientation error in all three states each	0.01%
Angular velocity error in all three states each	0.01%



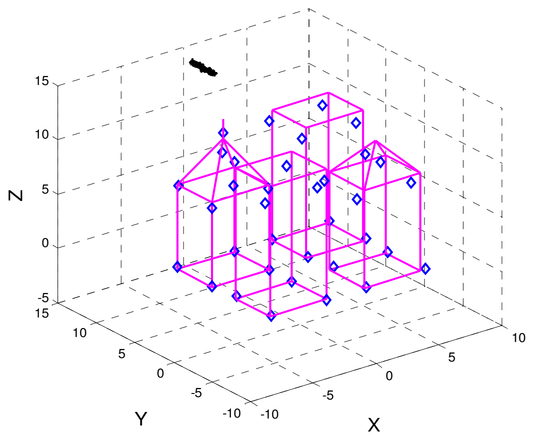
a) At time: 0 sec



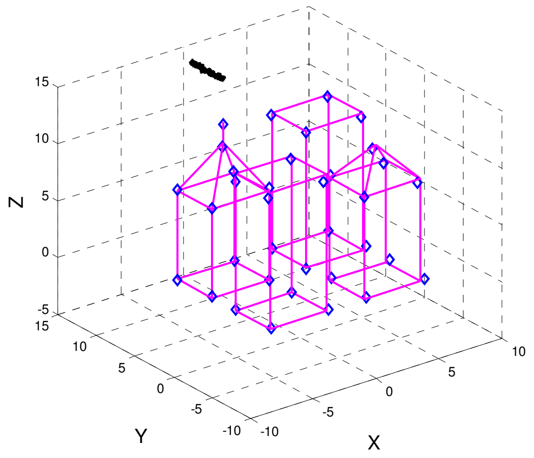
b) At time: 25 sec



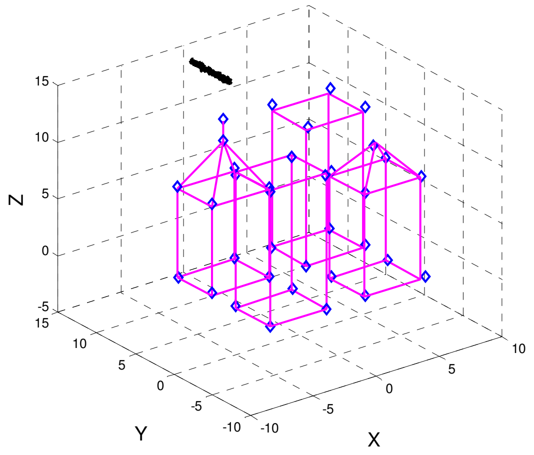
c) At time: 50 sec



d) At time: 75 sec



e) At time: 100 sec



f) At time: 125 sec

Figure 5.1: Forward Flight Simulation Results With 35 Feature Points. Image processing is updated at 10 frames/sec and UAV is flying forward at 1.4ft/sec. Convergence is good at 125 sec.

Hence, the results have shown that

- The proposed algorithm can successfully generate a 3D model of the scene, from 2D image information while flying forward towards the obstacle.
- The speed of flight is critical, as with too high a speed, obstacles will overlap the FOE. Too low a speed, on the contrary, will give very less new information for the update. Successful 3D modeling will not be possible in both such cases.
- Comparing the results of lateral flight simulations (Chapter 4) and the results presented here, it can be said that, flight duration required to generate a 3D model of the scene while flying forward, was 25% more than the duration of flight required for lateral flight case.
- Subject to the two conditions of features not exactly at FOE and using EKF for non-linearities, the simulation results show that for practical fields of view, the depth extracted from forward motion is indeed usable for a large part of the image.

CHAPTER 6

ANALYSIS OF RESULTS

Computational Effort

It may be realized that for detecting 'N_X' number of feature points (in 3D space), we have N_X* N_Z correspondences to be established, where the N_Z is the number of feature points picked up (observed) in every image. This holds for every iteration except the first one, when the database is empty and there is no correspondence to be done. So if the frame rate is 'f' frames per second and the simulation gives satisfactory results after 't' seconds, then the total number of images we use in the simulation are

$$\text{Number of images utilized: } N_I = f{*}t$$

Hence the total number of correspondences, the algorithm has to establish is given by

$$\text{No of Correspondences} = N_X{*}N_Z{*}N_I - 1$$

It may be realized that as the number of feature points in the scene increases (i.e. as N_Z increases), the number of points in the database N_X also increases accordingly (as the scene feature points eventually end up as points in the database, once the correspondence is established and 3D coordinates are found). Hence the computational effort increases tremendously, which in turn results in a need for much more simulation time, as well as, many more number of images required, in order to satisfactorily do the 3D obstacle detection/modeling. For the simulations above, the computational effort increased by about 14.7 times with an increase in number of feature points by 4.4 times (precisely from 4.054 seconds required to simulate 8 feature points vis-à-vis 60.201 seconds required to detect 35 feature points to within 2.5% of accuracy) for later flight. This further increased by yet another 25% for forward flight.

Image Acquisition / Frame rate

For the above analysis, a frame rate of 10 frames per second was supposed. Hence if the simulation ran for 60 seconds, 600 images were used (case of Fig 4 above). If a better frame grabber / image processor is available so that, about 30 frames per second frame rate is available, the simulation time will reduce to 20 seconds, which in turn means, lesser duration of lateral flight required, to correctly model the scene.

Error Analysis

Figure 6.1 presents with error analysis, for the simulation results presented in Figures 4.5, 4.6 and 5.1. One sample point randomly has been chosen for each of the three cases. All the three estimated coordinates of the selected feature points in the database has been compared with the actual value of coordinates. The results in fig 6.1 show that all cases converge to the actual point locations, to within about $\pm$3%. This indicates successful convergence of features points to their actual locations in 3D space.

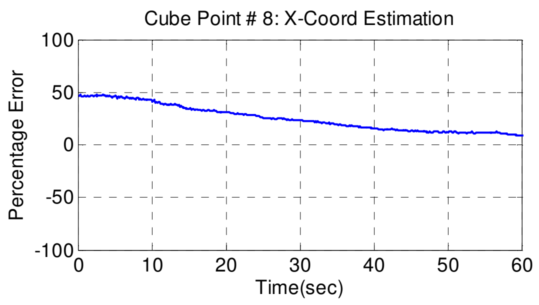
In the algorithm however, simulation is stopped when the average error from each of the three coordinates from all the feature points in the scene are successfully modeled (to within about $\pm$3% of the actual locations in 3D space). This means that for a case of 35 feature points, there are 35x3=105 coordinates to be estimated correctly.

Merits & Constraints of proposed Approach

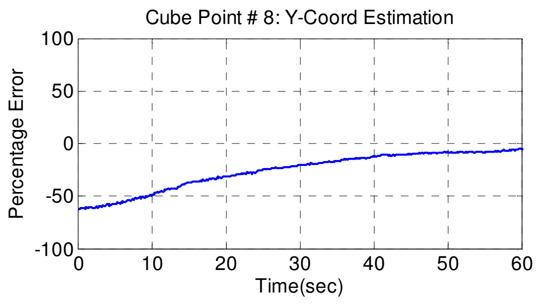
It may be said that the obvious merit of this approach in both lateral and forward flight, is that of providing a capability of 3D obstacle detection and modeling by using only one camera. This is of significant importance for future miniature UAVs, which might not be capable of carrying any other sensor, except for a single camera. The information that is obtained as a result of this algorithm, is that of a full scale 3D model of the scene, which may be directly utilized for any mission planning, as desired. On the contrary, the algorithm has an obvious constraint of tremendous increase in computational effort with an increase in number of feature points.

Further, the lateral flight pattern for such obstacle detection may also seem as a constraint, at least to a mission, which was that of moving forward towards the goal/target. The forward flight does overcome this constraint but increases computational effort by another 25%.

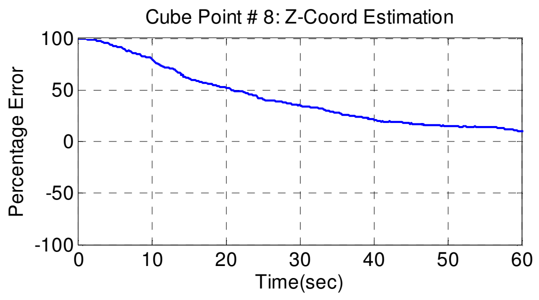
Yet another constraint is that of at least having some feature points at all, in the scene. If the UAV takes-off e.g in front of a flat wall, there are hardly any feature points to be detected and modeled, even with a lateral flight path. Such a problem, in fact, is a recommended subject of future work, as discussed in Chapter 7.



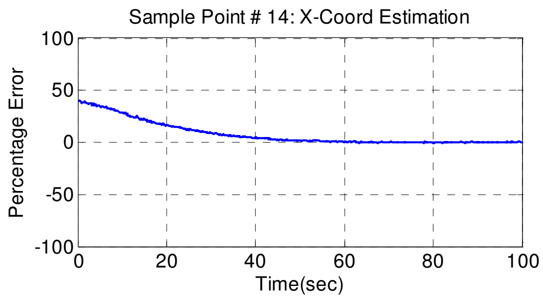
a)



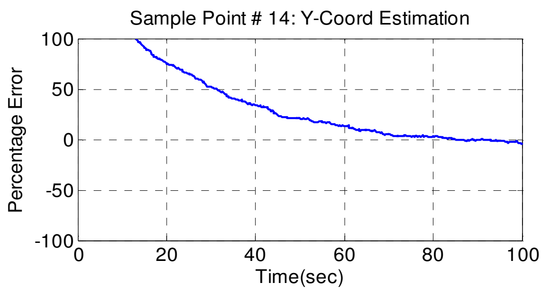
b)



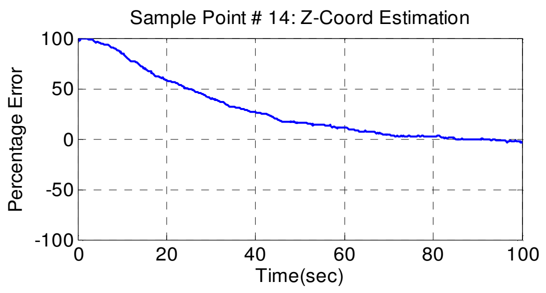
c)



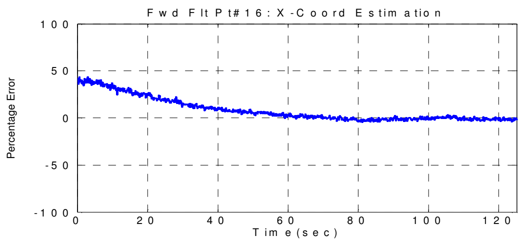
d)



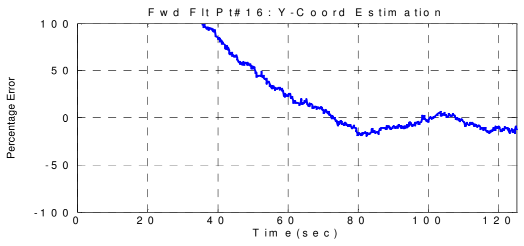
e)



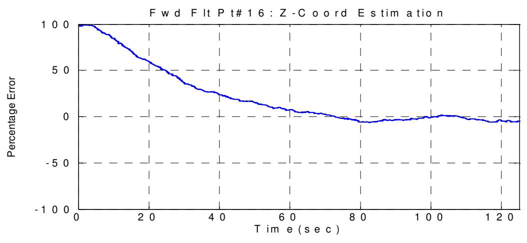
f)



g)



h)



i)

Figure 6.1: Error percentage versus Time, for Sample Points. *Plots a), b) and c) show estimation error progression in X, Y & Z coordinates, respectively, for a sample point from figure 4.5. Plots d), e) & f) are similar plots for a sample point from figure 4.6, whereas plots g), h) & i) are for figure 5.1 upto 100seconds.*

CHAPTER 7

CONCLUSION AND FUTURE WORK

Conclusion

From the proposed algorithm and associated simulations it is concluded that,

- The proposed algorithm can successfully generate a 3D model of the scene, from 2D image information.
- This modeling only requires one camera as the sensor.
- The results have been achieved for an unknown world and no constraints were put on the environment being modeled. No attributes of the environments were provided to the system, except for the 2D images being captured by the camera.
- The 3D scene model gives information of size and location of all obstacles in the scene. This information is sufficient to initiate an obstacle avoidance maneuver in 3D space.
- In the case of lateral flight the scene modeling has been achieved (to within $\pm$3% of actual 3D locations of the feature points) in 60 seconds of flight for 8 feature points, and 100 seconds of flight for 35 feature points.
- The successful 3D scene modeling required flying through a very small arc of lateral flight as compared to the size of object being modeled. There had been no need to capture images from all sides of the objects being modeled. Thus the approach is much better as compared to a typical 'Structure from Motion'

problem, which requires right, left, top or other views of the object, in order to generate its 3D model.

- The algorithm does require some feature points in the scene. Hence if no feature points are detected in the scene, the algorithm implies that there are no obstacles to be avoided and the initial flight path of the UAV may be continued without any disruption. This is almost always true in real world scenarios. However, there could be one exception of that of a flat wall in front, which is discussed as part of the future work.

Future Work

In future, it is planned to integrate some of the single sensor based approaches similar to Flow Field Divergence concept and/or Optical flow concept (as referred in Chapter 3 above) with the algorithm proposed in this paper. This is because each of the individual single sensor based approaches, has some constraints (e.g. of feature points, texture, flight path etc). Also some approaches tend to be computationally very heavy. Hence integration of more than one approaches, would be investigated for real time implementation.

An extreme case, that may be unsolvable by any of the approaches above, could be that of a flat wall in front. Since this would have no feature points, textures, optical flows or motion flow vectors, such a problem would not be handelable by any of such single sensor based approaches. For such a case a non-scanning (and hence a low weight) laser range finder may also be integrated with this set-up. The final approach is planned to be implemented on GTMax UAV.

APPENDIX A

DERIVATION OF EQUATIONS USED IN CHAPTER 4

Derivation of Equation 4.19

Referring equations 4.7 and 4.11, the residual vector on 2D image plane is given by

$$d = \begin{bmatrix} dx \\ dy \end{bmatrix} = Z - x_k = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} - \begin{bmatrix} y_k \\ z_k \end{bmatrix} \tag{A.1}$$

where $\boldsymbol{Z}$ is observed feature point on image plane and $\boldsymbol{x_k}$ is the projected feature point from 3D space on 2D image plane.

Hence,

$$\frac{\partial d}{\partial x_k} = \begin{bmatrix} \frac{\partial dx}{\partial y_k} & \frac{\partial dx}{\partial z_k} \\ \frac{\partial dy}{\partial y_k} & \frac{\partial dy}{\partial z_k} \end{bmatrix}$$

$$= \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} = -I_2$$

This is one of the results as used in Equation 4.19.

The second result is derived as follows.

Referring equation 4.8, the relative position vector of the camera is

$$\boldsymbol{X = X_v - X_p}$$

$$X = \begin{bmatrix} X_{ck} \\ Y_{ck} \\ Z_{ck} \end{bmatrix} = \begin{bmatrix} X_v \\ Y_v \\ Z_v \end{bmatrix} - \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix}$$

Referring equation 4.11, the corner feature projected onto 2D image place from 3D space is given by

$$x_k = \begin{bmatrix} y_k \\ z_k \end{bmatrix} = \frac{f}{X_{ck}} \begin{bmatrix} Y_{ck} \\ Z_{ck} \end{bmatrix} = \begin{bmatrix} Y_{ck} / X_{ck} \\ Z_{ck} / X_{ck} \end{bmatrix}$$

where unit focal length is supposed without loss of generality, the small letters indicate 2D coordinates in image plane and capital letters indicate coordinates in 3D world.

Hence,

$$\frac{\partial x_k}{\partial X} = \begin{bmatrix} \frac{\partial y_k}{\partial X_v} & \frac{\partial y_k}{\partial Y_v} & \frac{\partial y_k}{\partial Z_v} \\ \frac{\partial z_k}{\partial X_v} & \frac{\partial z_k}{\partial Y_v} & \frac{\partial z_k}{\partial Z_v} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial\left(\frac{Y_v - Y_p}{X_v - X_p}\right)}{\partial X_v} & \frac{\partial\left(\frac{Y_v - Y_p}{X_v - X_p}\right)}{\partial Y_v} & \frac{\partial\left(\frac{Y_v - Y_p}{X_v - X_p}\right)}{\partial Z_v} \\ \frac{\partial\left(\frac{Z_v - Z_p}{X_v - X_p}\right)}{\partial X_v} & \frac{\partial\left(\frac{Z_v - Z_p}{X_v - X_p}\right)}{\partial Y_v} & \frac{\partial\left(\frac{Z_v - Z_p}{X_v - X_p}\right)}{\partial Z_v} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{-(Y_v - Y_p)}{(X_v - X_p)^2} & \frac{(X_v - X_p)}{(X_v - X_p)^2} & 0 \\ \frac{-(Z_v - Z_p)}{(X_v - X_p)^2} & 0 & \frac{-(X_v - X_p)}{(X_v - X_p)^2} \end{bmatrix}$$

$$= \frac{1}{X_{ck}} \begin{bmatrix} \frac{-Y_{ck}}{X_{ck}} & 1 & 0 \\ \frac{-Z_{ck}}{X_{ck}} & 0 & 1 \end{bmatrix}$$

$$= \frac{1}{X_{ck}} \begin{bmatrix} -x_k & I_2 \end{bmatrix}$$

which is the required result as used in equation 4.19.

Derivation for Equations 4.21 and 4.22

Referring equation A.1 given above,

$$d = \begin{bmatrix} dx \\ dy \end{bmatrix} = Z - x_k = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} - \begin{bmatrix} y_k \\ z_k \end{bmatrix}$$

$$\frac{\partial d}{\partial Z} = \begin{bmatrix} \frac{\partial dx}{\partial z_1} & \frac{\partial dx}{\partial z_2} \\ \frac{\partial dy}{\partial z_1} & \frac{\partial dy}{\partial z_2} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= I_2$$

which is the required result.

Derivation for Equations 4.23 and 4.24

Referring equations 4.5 and equation A.1

$$J = dx^2 + dy^2$$

$$d = \begin{bmatrix} dx \\ dy \end{bmatrix}$$

$$\frac{\partial J}{\partial d} = \begin{bmatrix} \frac{\partial J}{\partial dx} & \frac{\partial J}{\partial dy} \end{bmatrix}$$

$$= \begin{bmatrix} 2dx & 2dy \end{bmatrix} = 2\begin{bmatrix} dx \\ dy \end{bmatrix}^T$$

$$= 2d^T$$

which is the required result as used in equations 4.23 and 4.24

REFERENCES

[1] Marr, D., *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, Freeman, New York, 1982.

[2] Wehner, R., “The polarization-vision project: Championing organisms biology”, *K.Schildberger and N.Elsner, Editiors, Neural Basis of Behavioural Adaptations,* Pages 103-143, Gustav Fischer Verlag, Stuttgart, 1994.

[3] Tsai, R., "A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology using Off-the-shelf TV Cameras and Lenses," *IEEE Trans. on Robotics and Automation*, Vol. 3, No.4, August 1987, Pg: 323- 344

[4] Johnson, E. N., and Schrage, D. P., “System Integration and Operation of a Research Unmanned Aerial Vehicle,” *VAIAA Journal of Aerospace Computing, Information, and Communication,* Vol.1, No.1, Jan. 2006.

[5] Ha, J.C., Johnson, E.N., and Tannenbaum, A.R., “Real-Time Visual Tracking System Using Active Contours for Navigation and Control of UAVs,” *Proceedings of the American Control Conference*, July 2007.

[6] Hayter, A. J., *Probability and Statistics for Engineers and Scientists.* 2nd Edition, Duxbury Pr, 2001.

[7] Watanabe, Y., Johnson, E.N., and Calise, A.J., “Vision-Based Approach to Obstacle Avoidance,” *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, August 2005.

[8] Gonzalez, R. C., Woods, R. E., Eddins, S. L., *Digital Image Processing Using Matlab*^{R} , Pearson Prentice-Hall, 2004.

[9] Grunewald, M., and Sitte, J., “A Resource Efficient Approach to Obstacle Avoidance via Optical Flow”, *Proceedings of 5th International Heinz Nisdorf Symposium: Autonomous Minirobots for Research and Edutainment (AMIRE), volume 97 of HNI-Verlagsschriftenreihe*, pages 205-214, Heinz Nixdorf Institute, October 2001.

[10] Watanable, Y., Johnson, E. N., and Calise, A. J., "Stochastically Optimized Monocular Vision-Based Guidance Design", *AIAA Guidance, Navigation, and Control Conference,* 2007, p 5349-5364.

[11] Coombs, D., Roberts, K., "Bee-Bot: Using Peripheral Optical Flow to Avoid Obstacles" *Intelligent Robots and Computer Vision XI*, , SPIE Vol 1825 1992.

[12] Argyros, A., A., and Bergholm, F., "Combining Central and Peripheral Vision for Reactive Robot Navigation" *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pages II:646-651, IEEE Computer Society Press, 1999.

[13] Takeno, J., Shinogi, Y., Nishiyama, S., Mizuguchi, N., Sorimat, K., "Realization of a 3D Vision Mobile Robot that can Avoid Collision with Moving Obstacles," *IEEE International Conference on Robotics and Automation,* Sacramento, California April 1991.

[14] Langer, D., and Jochem, T., "Fusing Radar and Vision for Detecting, Classifying and Avoiding Roadway Obstacles", *IEEE Intelligent Vehicles Symposium,* 1996 pp333-338.

[15] Ilic, M., Masciangelo, S.; Pianigiani, E., "Ground plane obstacle detection from optical flow anomalies: A robust and efficient implementation*", IEEE Intelligent Vehicles Symposium, Proceedings, 1994,* p 333-338.

[16] Mikawa, M., Yoshida, K., Tanno, M., and Matsumoto, M., "Vision-based redundancy control of robot manipulators for obstacle avoidance", *IECON Proceedings (Industrial Electronics Conference)*, v 3, 1997, p 1373-1378.

[17] Lenser, S., Veloso, M., "Visual Sonar: Fast Obstacle Avoidance Using Monocular Vision", *IEEE International Conference on Intelligent Robots and Systems*, v 1, 2003, p 886-891.

[18] Nelson, R. C., "Visual Navigation", PhD Thesis, University of Maryland, August 1988.

[19] Nelson, R. C., Aloimonos, J., "Using flow field divergence for obstacle avoidance towards qualitative vision", *Second International Conference on Computer Vision*, (IEEE Cat No 88Ch2664-1) 1988, p 188-96.

[20] Nelson, R. C., Aloimonos, J., “Obstacle Avoidance Using Flow Field Divergence”, *IEEE Transactions On Pattern Analysis And Machine Intelligence,* Vol 11 No 10 October 1989.

[21] Young, G. S., Hong, T. H., Herman, M., Yang, J. C. S., “Obstacle detection for a vehicle using optical flow”, *Proceedings of the Intelligent Vehicles '92 Symposium (Cat. No.92TH0468-9),* 1990, p 185-90.

[22] Hatsopoulos, N., Anderson, J.A., “Collision-avoidance system based on optical flow”, *Proceedings of the Intelligent Vehicles '92 Symposium (Cat. No.92TH0468-9),* 1990, p 79-84.

[23] Nakao, K., Kondo, K., Kobashi, S., Hata, Y., Yagi, T., “Shape Reconstruction Using Extended Kalman Filter with an Active Camera”, *IEEE International Symposium on Communications and Information Technologies 2004 (IEEE Cat. No.04EX897)*, Oct 2004, Japan, p 857-60 vol.2.

[24] Watanabe, Y., “Stochastically Optimized Monocular Vision-Based Navigation and Guidance”, PhD Thesis, Georgia Institute of Technology, April 2008.

[25] Matthies, L., Kanade, T., Szeliski, R., “Kalman Filter-Basd Algorithms for Estimating Depth from Image Sequences”, *International Journal of Computer Vision,* v3, n3, 1989, p209-238.

[26] Hirokawa, R., Kubo, D., Suzuki, S., Meguro, J., Suzuki, T., “A Small UAV for Immediate Hazard Map Generation”, *AIAA Infotech@Aerospace Conference and Exhibit,* May 2007, p 153-158.