TRACKING AND NAVIGATION METHODS FOR UNMANNED AERIAL VEHICLES

Sarjit Goswami
B.E., Gujarat University, India, 2008

PROJECT

Submitted in partial satisfaction of the requirements for the degree in

MASTER OF SCIENCE

in

ELECTRICAL AND ELECTRONIC ENGINEERING

at

CALIFORNIA STATE UNIVERSITY, SACRAMENTO

SUMMER
2012
TRACKING AND NAVIGATION METHODS FOR UNMANNED AERIAL VEHICLES

A Project

by

Sarjit Goswami

Approved by:

__________________________________, Committee Chair
Fethi Belkhouche, Ph.D.

__________________________________, Second Reader
Preetham B. Kumar, Ph.D.

Date
Student: Sarjit Goswami

I certify that this student has met the requirements for format contained in the University format manual, and that this project is suitable for shelving in the Library and credit is to be awarded for the Project.

__________________________, Graduate Coordinator
Preetham B. Kumar, Ph.D.

__________________________, Date

Department of Electrical and Electronic Engineering
This project deals with tracking and navigation methods for Unmanned Aerial Vehicles (UAVs). Tracking, navigation, and online flight path control are essential for any application employing UAVs. Issues such as optimal flight path and obstacle avoidance are also important for the deployment of UAVs in real world applications. This project employs plane decomposition and kinematics-based rules to derive linear navigation control laws. These control laws depend on navigation parameters such as proportionality and deviation factors used to guide the UAV to successfully reach a moving goal following a non-linear path. Two modes of UAV motion are presented in the project: 1) Surveillance mode in which a UAV follows a moving goal in a three dimensional space and eventually reaches it with the help of the navigation law. 2) Obstacle avoidance mode in which the UAV maneuvers around the obstacles in its flight path and reaches the
moving goal with the help of the navigation law. Through simulation, it is shown that the navigation laws used in this project derived based on kinematics rules are adequate for successful implementation of the tracking and navigation of the UAV, allowing the vehicle to reach an arbitrary goal moving with nonlinear maneuvers.

_______________________, Committee Chair
Fethi Belkhouche, Ph.D.

_______________________
Date
ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all the people, academic and otherwise who have been extremely supportive and helpful not only limited to this project but also throughout my grad school years. First and foremost, I’d like to express my utmost gratitude to Dr. Fethi Belkhouche, my project advisor for always being extremely helpful and patient. Thank you for all your support, guidance, help and most of all for being patient. And thank you for providing me an opportunity to be able to work with someone as amazing as yourself. Your selfless help and support has a meant a lot to me over the years and has finally resulted in me being able to finish this project successfully. Thank you Dr. Preetham Kumar for being ever present and willing to solve any problem I faced during my entire Masters program. I express my utmost gratitude for all the invaluable academic help that you’ve provided me over the years. Thank you all the faculty in the Electrical & Electronics Engineering Department for the invaluable knowledge that you’ve imparted me for which there is no parallel. A big thank you for the staff as well for always being kind and cooperative.

As for the non academic people who have been very important and extremely supportive during my journey so far I’d like to express my unbridled love and gratitude to my family who’ve always believed in me and loved me no matter what. Thanks a ton mum n dad, I’m nothing without you. Not to mention all my wonderful friends who’ve always been there to uplift my spirits and provide encouragement when it was most required. My utmost gratitude and regards to Ahsan Syed and Hammad Shah – my roommates and brothers. Thanks for tolerating me, you guys know you’re the best! Avinash Pandey and Ayush Chadha for always being very compensating, kind and motivating. You guys deserve kudos – kindly applaud for yourselves. My
friends Armen Nalbandyan, Chris Harrington and Gabriel Hojman for their awesome company and all the good times we’ve had so far. Those are memories that will never die. You guys always resuscitated me in one way or another and that’s why you all are awesome and will always be. And all other friends I couldn’t mention for the lack of space – God bless you all and thanks for the awesome company.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements ........................................................................................................ vi</td>
</tr>
<tr>
<td>List of Figures .................................................................................................................. x</td>
</tr>
<tr>
<td>Chapter</td>
</tr>
<tr>
<td>1. INTRODUCTION .................................................................................................................... 1</td>
</tr>
<tr>
<td>2. TRACKING AND NAVIGATION METHODS FOR UNMANNED AERIAL VEHICLES ................................................................. 4</td>
</tr>
<tr>
<td>2.1 UAV navigation using vision sensors .............................................................................. 4</td>
</tr>
<tr>
<td>2.2 UAV navigation using inertial sensors ........................................................................... 6</td>
</tr>
<tr>
<td>2.3 UAV navigation using integrated sensor systems ......................................................... 7</td>
</tr>
<tr>
<td>2.4 UAV navigation using hierarchical control ................................................................. 7</td>
</tr>
<tr>
<td>2.5 UAV navigation using feedback linearization .............................................................. 9</td>
</tr>
<tr>
<td>3. KINEMATICS-BASED NAVIGATION LAWS AND PLANE DECOMPOSITION ................................. 10</td>
</tr>
<tr>
<td>3.1 Definitions and kinematics modeling of UAV motion ................................................... 10</td>
</tr>
<tr>
<td>3.2 Definition of the flight path, heading and line of sight angles ..................................... 12</td>
</tr>
<tr>
<td>3.3 Plane decomposition method ........................................................................................ 13</td>
</tr>
<tr>
<td>3.4 Line of sight angles in the (x,y) and (r,z) planes ......................................................... 15</td>
</tr>
<tr>
<td>3.5 Kinematics equations and dynamic constraints ........................................................... 16</td>
</tr>
<tr>
<td>3.6 Control law for flight path angle and heading angle .................................................... 18</td>
</tr>
</tbody>
</table>
4. OBSTACLE AVOIDANCE .................................................................................................................. 20
   4.1 Zero slope deviation and infinite slope deviation ................................................................. 22
   4.2 Derivation of the new navigation parameters for obstacle avoidance .......................... 24
   4.3 Examples of obstacle avoidance ...................................................................................... 25

5. SIMULATION RESULTS FOR SURVEILLANCE AND OBSTACLE AVOIDANCE ................................................................. 28
   5.1 Simulation results for surveillance mode ........................................................................ 28
   5.2 Simulation results for obstacle avoidance mode .............................................................. 38

6. CONCLUSION .................................................................................................................................. 45

Appendix MATLAB Code ............................................................................................................... 47

References ......................................................................................................................................... 57
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Results obtained from a visual sensor (a) original image (b) optimization</td>
<td>5</td>
</tr>
<tr>
<td>2.</td>
<td>Layout of an Inertial Navigation System used in Quad rotor UAV</td>
<td>6</td>
</tr>
<tr>
<td>3.</td>
<td>Hierarchical control architecture used in an autonomous helicopter</td>
<td>8</td>
</tr>
<tr>
<td>4.</td>
<td>Representation of UAV motion in 3 dimensional space</td>
<td>11</td>
</tr>
<tr>
<td>5.</td>
<td>Representation of the flight path angle, the heading angle and the line of sight angles</td>
<td>12</td>
</tr>
<tr>
<td>6.</td>
<td>Decomposition of 3 dimensional space into two 2 dimensional planes: (x,y) and (x,z)</td>
<td>13</td>
</tr>
<tr>
<td>7.</td>
<td>Line of sight representation from robot to goal</td>
<td>15</td>
</tr>
<tr>
<td>8.</td>
<td>Obstacles in the (x,y) and (r,z) planes</td>
<td>21</td>
</tr>
<tr>
<td>9.</td>
<td>Motion of robot in the horizontal plane in the presence of obstacles (a) without deviation – collision occurs (b) with zero slope deviation – collision avoided</td>
<td>26</td>
</tr>
<tr>
<td>10.</td>
<td>Motion of robot in the vertical plane in presence of obstacles (a) without deviation – collision occurs (b) with infinite slope deviation – collision avoided</td>
<td>27</td>
</tr>
<tr>
<td>11.</td>
<td>The initial position of robot (red dot) and target (green dot) at time t=0 in the 3 dimensional space</td>
<td>28</td>
</tr>
<tr>
<td>12.</td>
<td>Initial position of robot (red star) and target (green star located on extreme left bottom) in the horizontal (x,y) plane at time t=0</td>
<td>29</td>
</tr>
<tr>
<td>13.</td>
<td>Initial position of the robot (red star) and target (not visible at time t=0 due to the constraints provided) in the vertical (r,z) plane at time t=0</td>
<td>29</td>
</tr>
<tr>
<td>14.</td>
<td>The position of the robot (red dot) and the target (green dot) after 5 steps in the 3 dimensional space</td>
<td>30</td>
</tr>
<tr>
<td>15.</td>
<td>The position of the robot (red star) and the target (green star) after 5 steps in the horizontal (x,y) plane</td>
<td>30</td>
</tr>
</tbody>
</table>
16. The position of the robot (red star) and the target (green star) after 5 steps in the vertical (r,z) plane……………………………………………………………………..31

17. The position of the robot (red dot) and the target (green dot) after 10 steps in the 3 dimensional space…………………………………………………………………..31

18. The position of the robot (red star) and the target (green star) after 10 steps in the horizontal (x,y) plane……………………………………………………………………..32

19. The position of the robot (red star) and the target (green star) after 10 steps in the vertical (r,z) plane……………………………………………………………………..32

20. The position of the robot (red dot) and the target (green dot) after 15 steps in the 3 dimensional space……………………………………………………………………..33

21. The position of the robot (red star) and the target (green star) after 15 steps in the horizontal (x,y) plane……………………………………………………………………..33

22. The position of the robot (red star) and the target (green star) after 15 steps in the vertical (r,z) plane……………………………………………………………………..34

23. The position of the robot (red dot) and the target (green dot) after 30 steps in the 3 dimensional space……………………………………………………………………..34

24. The position of the robot (red star) and the target (green star) after 30 steps in the horizontal (x,y) plane……………………………………………………………………..35

25. The position of the robot (red star) and the target (green star) after 30 steps in the vertical (r,z) plane……………………………………………………………………..35

26. The position of the robot (red dot) and the target (green dot) after 35 steps in the 3 dimensional space……………………………………………………………………..36

27. The position of the robot (red star) and the target (green star) after 35 steps in the horizontal (x,y) plane……………………………………………………………………..36

28. The position of the robot (red star) and the target (green star) after 35 steps in the vertical (r,z) plane……………………………………………………………………..37

29. The initial position of the robot (red dot) and the target (green dot) at t=0 in the 3 dimensional space……………………………………………………………………..38
30. The initial position of the robot (red star) and the target (green star located at top right corner) at time t=0 in the horizontal (x,y) plane………………………………………..38

31. The initial position of the robot (red star) and target (not visible at time t=0 due to the constraints provided) at time t=0 in the vertical (r,z) plane……………………………39

32. The position of the robot (red dot) and the target (green dot) after 15 steps in the 3 dimensional space……………………………………………………………………39

33. The position of the robot (red star) and the target (green star) after 15 steps in the horizontal (x,y) plane………………………………………………………………………40

34. The position of the robot (red star) and the target (green star) after 15 steps in the vertical (r,z) plane………………………………………………………………………40

35. The position of the robot (red dot) and the target (green dot) after 22 steps in the 3 dimensional space………………………………………………………………………41

36. The initial position of the alternative path followed by the robot (red dot) and the target (green dot) on encountering an obstacle………………………………………41

37. The position of the robot (red star) and the target (green star) after 22 steps in the horizontal (x,y) plane………………………………………………………………………42

38. The position of the robot (red star) and the target (green star) after 22 steps in the vertical (r,z) plane………………………………………………………………………42

39. The position of the robot (red dot) and the target (green dot) after 50 steps in the 3 dimensional space………………………………………………………………………43

40. The position of the alternative path followed by the robot (red dot) and the target (green dot) after 28 steps on encountering an obstacle…………………………………………43

41. The position of the robot (red star) and the target (green star) after 50 steps in the horizontal (x,y) plane………………………………………………………………………44

42. The position of the robot (red star) and the target (green star) after 50 steps in the vertical (r,z) plane………………………………………………………………………44
Chapter 1

INTRODUCTION

Unmanned Aerial Vehicles more commonly known by their acronym UAV are essentially autonomous mobile robots used in aerial environments. The name “Unmanned Aerial Vehicles” suggests that these robots are operated by preprogrammed algorithms and therefore require little or no human help during operation. The algorithms are translated to a smart code which is able to direct the UAV dynamically to follow a given desired path to attain a specified goal with varying levels of path selection as well achieving the objective of obstacle avoidance. Online path planning (in flight path modification) is an essential and sensitive feature required in UAV flight motion in order to dynamically achieve the desired UAV action.

There are various methods employed for UAV path planning and navigation. The most common actions employing UAVs include reaching a stationary goal, reaching a moving goal or following a moving goal at constant distance. Depending on the desired action there are different navigation methods. The most common methods used for the purpose include vision based sensors, audible sensors, GPS navigation system, GPS/INS integration and integrated sensor system among others. Each of these methods represent a solution specific to the task assigned to the UAV in a particular scenario.
The UAV navigation method presented in this project demonstrates the use of robust kinematics based navigation laws derived using control theory, where linear navigation techniques as well as variable navigation parameters provide an optimal way to generate non straight paths to the goal. The two cases discussed in this project are: 1) UAV following a moving goal in a three dimensional space with the objective of reaching the goal using the navigation law presented in the project. 2) UAV following a moving goal in a three dimensional space in the presence of obstacles in the path. In this case, the ability of the UAV to maneuver around the obstacles successfully in order to reach the desired goal is an important feature.

Another important aspect discussed in this project deals with the concept of plane decomposition wherein a three dimensional space is divided into two 2 dimensional planes for the purpose of simplifying the process of flight path navigation. The derivation of the navigation law as well as a complete terminology used therein are discussed in detail in order to explain the complete process of UAV navigation and flight path control.

The “Obstacle Avoidance Mode” for the UAV is discussed in detail in order to explain the ability of the UAV to maneuver around the obstacles present in the path and successfully reach the goal using an alternate path by deviating towards waypoints. This is achieved successfully by modifying the navigation parameters (proportionality and
deviation factors) in the two dimensional planes and with appropriate assumptions made for the waypoints in both cases.

MATLAB is used as a tool of choice for programming the algorithms and simulating the methods used for flight path generation and navigation. MATLAB code used to simulate the methods for both cases is presented in the appendix of the report. Simulations are included in Chapter 5. The simulation shows the successful implementation of the kinematics-based navigation law to achieve the desired UAV actions of reaching a moving goal both with and without obstacles.
The various methods used for UAV tracking and navigation can be classified broadly into two categories: 1) Sensor based navigation methods and 2) Model based navigation methods [1]. Various types of sensors are commonly used for the purpose of UAV path planning and navigation. The most commonly used types of sensors include vision based, inertial and sonar sensors [1]. In cases requiring a greater degree of precision, a combination of different sensors is used which provides a greater flexibility and better results. The different model based navigation techniques commonly employed include hierarchical control, feedback linearization, state-dependent Riccati equation, H-infinity theory, Lyapunov theory, and adaptive network-based constrained optimization [1]. This chapter sheds light on some of the most commonly used methods for UAV path planning and navigation which include vision based sensors, inertial sensors, integrated sensor systems, hierarchical control and feedback linearization.

2.1 UAV navigation using vision sensors

The simple problem of horizon detection by a UAV shows a simple example that displays the functionality of vision based sensors. The basic function of a UAV in such a case is to distinguish between the ground and the horizon using visual sensors. Even though this
task sounds simple, it employs highly advanced techniques such as rudimentary image segmentation, real-time optimization, elementary multilevel filtering and feedback stabilization [2]. An example of the functionality of the method is displayed in the following figure:

Figure 1 – Results obtained from a visual sensor [2] (a) original image (b) optimization

The figure above displays how segmentation and differentiation between ground and sky is obtained with the help of feedback stabilization techniques from control theory [2]. The details of control theory based concepts used in determining the function of vision sensors are beyond the scope of this project, the example serves the purpose of illustrating the benefits of vision based sensors used in UAV path planning and navigation.

However, vision based sensors are not always the best bet in achieving desired UAV path planning and navigation results. For example, in the horizon detection problem presented above, if the horizon is not a clearly distinguishable line the optical results obtained from
the visual sensors may exhibit a certain discrepancy which may be undesired. Moreover, the synthesis of millions of pixels in real time may be impractical.

2.2 UAV navigation using inertial sensors

Inertial Navigation Systems (INS) or Inertial Measurement Units (IMU) are the most common forms of inertial sensors used in UAV navigation. The advantage of inertial sensors over other sensors is that they tend to operate independently of external factors such as wind direction, wind force, friction, interference etc. A common form of INS used in UAVs is described in the following figure:

![Figure 2 – Layout of an Inertial Navigation System used in Quad rotor UAV [3]](image)

As shown in the figure, the INS unit is made up of commonly available physical components such as quad rotors, accelerometers, gyroscopes and microcontrollers. The
INS is connected to the main processor, which provides the main processor with sensing information. However the major disadvantage of Inertial Sensors or INS/IMU is that the position errors obtained during UAV tracking compound over time which yields erroneous results [3].

2.3 UAV navigation using integrated sensor systems

In order to compensate for the various errors encountered by a single type of sensors employed in UAV navigation, different kinds of sensors are fused or integrated into a single design to provide optimal results. Various kinds of combinations exist between different kinds of sensors such as vision/acoustic and vision/inertial [1].

2.4 UAV navigation using hierarchical control

One of the most common model-based approaches used for UAV navigation is the hierarchical control method in which a control system using a hierarchical organizational structure is used for the purpose of path planning. A simple hierarchical control system structure for an autonomous helicopter is as displayed below:
Figure 3 – Hierarchical control architecture used in an autonomous helicopter [4]

The figure above illustrates a hierarchical control system structure which integrates the Ground Control Unit along with the on board software which collectively provide the function of UAV path planning and navigation. As shown in the figure, both blocks are linked via a human operator. Various path control functions such as path estimation,
collision detection and avoidance, actual path control and navigation as well as the
dynamics of the vehicle are integrated into the on board software which plays a major
role in the path planning and navigation process [4]. Thus, the control system architecture
which employs a highly organized hierarchical design serves to provide optimal path
planning and navigation solution for the Unmanned Aerial Robot.

2.5 UAV navigation using feedback linearization

In order to address a UAV environment wherein a nonlinear dynamic motion is employed
the concept of dual controller is used. For example, feedback linearization is applied
within an inner controller and it is combined with a linear outer controller in order to
achieve robust navigation capabilities [5].

This chapter gives a brief overview of some of the most commonly used methods for
tracking and navigation of Unmanned Aerial Vehicles. Methods of UAV navigation
using vision sensors, inertial sensors, integrated sensor systems, hierarchical control and
feedback linearization are discussed with proper examples.
Chapter 3

KINEMATICS-BASED NAVIGATION LAWS AND PLANE DECOMPOSITION

This project employs a navigation law derived-based on the kinematics system of equations. The navigation law allows to successfully plan the path of the UAV so that it reaches the desired moving goal. The kinematics based control and navigation laws are advantageous compared to various sensor-based and model-based navigation methods discussed in chapter 2 as they eliminate some problems experienced with those methods and provide a reliable method for online path planning of the UAV. These laws utilize navigation parameters such as proportionality factors and deviation factors in order to achieve successful trajectory control of the UAV. This is another important advantage of these laws.

3.1 Definitions and kinematics modeling of UAV motion

The motion of a UAV moving in a three dimensional space ( (x,y,z) Cartesian co-ordinate system) can be described with the help of the following figure:
The terms appearing in the figure can be defined as:

robot : The UAV is represented by a sphere in the 3 dimensional space for simplicity.

\( \nu_r \) : The velocity of the robot in the 3 dimensional space.

\( \rho_r \) : The Euclidean distance from the origin to the robot in the 3 dimensional space.

\( x_r \) : The projection of \( \rho_r \) in the (x,y) plane.

\( x_r \) : The corresponding x co-ordinate indicating the x position of the robot at any given time in the 3 dimensional space.
\( y_r \): The corresponding y co-ordinate indicating the y position of the robot at any given time in the 3 dimensional space.

\( z_r \): The corresponding z co-ordinate indicating the z position of the robot at any given time in the 3 dimensional space.

### 3.2 Definition of the flight path, heading and line of sight angles

![Diagram showing flight path angles, heading angles, and line of sight angles.](image)

**Figure 5** – Representation of the flight path angle, the heading angle and the line of sight angles

With reference to figure 5, we define the following variables:

\( \varphi_r \): flight path angle (angle controlling the motion of the robot in vertical direction)

\( \psi_r \): heading angle (angle controlling the motion of the robot in horizontal direction)
\( \alpha_1, \alpha_2 \): line of sight angles of the robot with respect to origin in the horizontal and vertical planes (the flight path angle and heading angle tend to follow \( \alpha_1 \) and \( \alpha_2 \), respectively).

### 3.3 Plane decomposition method

In order to simplify the analysis and the mathematical derivation, the 3 dimensional work space is divided into two 2 dimensional planes: the (x,y) plane and the (x,z) plane [1]. The plane decomposition process can be explained using the following figure:

Figure 6 – Decomposition of 3 dimensional space into two 2 dimensional planes : (x,y) and (x,z)
Taking into account the values of $\rho_r$, $r_r$ from figure 4 as well as the values of the line of sight angles $\sigma_1$ and $\sigma_2$ defined in figure 5, the co-ordinates of the robot at any point in space can be defined by the following equations:

Co-ordinates in (x,y) plane:

$$x_r = r_r \cos(\sigma_2), \quad y_r = r_r \sin(\sigma_2)$$  \hspace{1cm} (1)

Co-ordinates in (r,z) plane:

$$r_r = \rho_r \cos(\sigma_1), \quad z_r = \rho_r \sin(\sigma_1)$$  \hspace{1cm} (2)

Thus, the co-ordinates of the robot in the original 3 dimensional workspace can be determined effectively using the plane decomposition method at any point in time. This approach considerably simplifies the problem of determining the position of the robot and therefore proven to be effective in the overall process of path planning and navigation.
3.4 Line of sight angles in the (x,y) and (r,z) planes

Figure 7 – Line of sight representation from robot to goal

Now that the planes have been defined by decomposing the 3 dimensional space, we define the line of sight angles for the respective planes. For this purpose we consider a stationary goal (for simplicity) and a robot moving towards the goal. The figure above describes the scenario aptly. Let L be the line of sight (imaginary line joining the moving robot and stationary goal). The direction of vector L is towards the goal from the robot [1]. The projection of vector L in the (x,y) plane is denoted by $L_{xy}$. The coordinates of the robot and the goal are defined as $(x_r, y_r, z_r)$ and $(x_g, y_g, z_g)$ respectively.
Now the line of sight angle for the horizontal (x,y) plane is defined as $\sigma_{xy}$ and for the vertical (x,z) plane is defined as $\sigma_{xz}$. Taking into consideration the coordinates of the robot and the goal, the line of sight angles in the respective planes can be defined as:

$$\tan \sigma_{xy} = \frac{y_{gr}}{x_{gr}} \quad \text{where} \quad y_{gr} = y_{g} - y_{r} \quad \text{and} \quad x_{gr} = x_{g} - x_{r}$$

$$\tan \sigma_{xz} = \frac{z_{gr}}{r_{gr}} \quad \text{where} \quad z_{gr} = z_{g} - z_{r} \quad \text{and} \quad r_{gr} = \sqrt{x_{gr}^2 + y_{gr}^2}$$

The equations above are valid for the case of a moving goal as well. In our assumed case, the goal isn’t moving therefore the coordinates of the goal will remain the same, however, the line of sight angles vary with time.

### 3.5 Kinematics equations and dynamic constraints

The kinematics equations defining the motion of the moving robot in a 3 dimensional space can be deduced based on the values of the various parameters discussed above. The kinematics equations of motion can be modeled as follows:

$$\dot{x}_r = v_r \cos(\varphi_r) \cos(\Psi_r)$$

$$\dot{y}_r = v_r \cos(\varphi_r) \sin(\Psi_r)$$

$$\dot{z}_r = v_r \sin(\varphi_r)$$
where, \( \dot{x}_r, \dot{y}_r \) and \( \dot{z}_r \) are the velocity components defining the motion of robot in the 3 dimensional space [1].

\( v_r \) is the velocity of the robot.

\( \varphi_r \) is the flight path angle (angle denoting motion in vertical plane)

\( \psi_r \) is the heading angle (angle denoting motion in horizontal plane)

For the sake of simplicity, the quantity \( v_r \cos(\varphi_r) \) can be denoted as \( v_1 \), resulting in the following kinematics equations in the \((x,y)\) plane

\[
\dot{x}_r = v_1 \cos(\psi_r) \\
\dot{y}_r = v_1 \sin(\psi_r)
\]

The dynamic constraints are an important aspect when discussing the motion of a vehicle in a 3 dimensional space as they characterize the limitations on angular velocity and the turning radius of the vehicle. The dynamic constraints in this case can be represented as

\[
0 < v_{\text{min}} \leq v_r \leq v_{\text{max}}
\]

\[
K_1 \leq \psi_r \leq K_1
\]

\[
K_2 \leq \varphi_r \leq K_2
\]

where, \( v_{\text{min}} \) and \( v_{\text{max}} \) are the minimum and maximum speeds of the robot respectively. \( K_1 \) and \( K_2 \) are constants which depend on the type of the vehicle and describe the limits on the angular velocities of the vehicle depending on the values of the flight path angle
and heading angle. Moreover, from (8) and (9) we conclude that \( \mathcal{V}_1 \geq 0 \) at all times which results in the values of flight path angle \( \Phi_r \) to always be within the range \([-\pi/2, \pi/2]\) [1].

### 3.6 Control law for flight path angle and heading angle

It is evident that the flight path angle and heading angle play the most important role in the guidance and navigation of the UAV. Therefore control laws for the flight path angle and heading angle are defined as follows:

\[
\Phi_r = N_1 \sigma_{xy} + c_1
\]

\[
\Phi_r = N_2 \sigma_{xz} + c_2
\]

where \( N_1 \) and \( N_2 \) are called proportionality factors, and \( c_1 \) and \( c_2 \) are called deviation factors. They are used to define curvature control of the robot. The proportionality factors and the deviation factors are collectively known as navigation parameters which essentially control the motion of the robot [1].

Different functionalities such as following a certain kind of path, avoiding obstacles and following an alternate path are provided by modifying the values of the navigation parameters. Therefore online path planning can effectively be achieved with the help of variable navigation parameters.
The most important condition for the robot to effectively perform is that \( N_1, N_2 \geq 1 \) [1].

This project utilizes different values of \( N_1 \) and \( N_2 \) to successfully generate path and avoid obstacles, which is discussed in the next chapter.

The desired values of the flight path angle ‘\( \phi_r \)’ and the heading angle ‘\( \psi_r \)’ and their initial values are not the same. In order to address this fact, a procedure known as “heading regulation” is employed [1]. Heading regulation procedure dictates that the final values of the flight path and heading angles are given as:

\[
\begin{align*}
\phi_r (t) &= N_1 \sigma_{xy} (t) + c_1 + b_1 e^{-d_1(t-t_0)} \\
\psi_r (t) &= N_2 \sigma_{rz} (t) + c_2 + b_2 e^{-d_2(t-t_0)}
\end{align*}
\] (15)

\[
\begin{align*}
\phi_r (t_0) &= N_1 \sigma_{xy} (t_0) + c_1 + b_1 \\
\psi_r (t_0) &= N_2 \sigma_{rz} (t_0) + c_2 + b_2
\end{align*}
\] (16)

The values of \( b_1 \) and \( b_2 \) are chosen so that the equations for the flight path and heading angle at initial time \( t_0 \) can be represented by:

\[
\begin{align*}
\phi_r (t_0) &= N_1 \sigma_{xy} (t_0) + c_1 + b_1 \\
\psi_r (t_0) &= N_2 \sigma_{rz} (t_0) + c_2 + b_2
\end{align*}
\] (17)

In (15) and (16) \( d_1 \) and \( d_2 \) are constants and satisfy \( d_1, d_2 > 0 \). They determine the turning radius of the vehicle. Smaller values of \( d_1 \) and \( d_2 \) result in wider turns and vice versa [1]. Thus the heading regulation procedure brings to light certain extra terms which are critical for UAV path planning and navigation.
A particular case for the UAV motion presented in this project would be when the robot moves towards a stationary or a moving goal and eventually reaches it without encountering any obstacles in the path. However, in practical applications, various kinds of obstacles and hindrances are present in the path of the robot. An essential part of this project is to address the problem of the obstacles present in the path of the robot so that it moves around the obstacles and successfully reaches the goal.

For simplicity, any obstacle in the path of the robot is treated as a sphere. Taking into account the method of plane decomposition discussed earlier, we use projections of the spherical obstacle in the 2 dimensional planes [1]. This results in the obstacle appearing as circles in the (x,y) and (r,z) planes. We denote the obstacles by $O_b$. Thus the obstacle in (x,y) plane is denoted as $O_{bxy}$ and the one in the (r,z) plane is denoted as $O_{brz}$. Under this assumption, the robot is considered to be a point and the obstacle is therefore a multiple of the radius of the robot.

The graphical representation of obstacles in both horizontal and vertical planes is as shown below:
Figure 8 (a) and (b) represent the obstacles in the (x,y) and (r,z) planes, respectively. Let us discuss both cases one at a time. In the (x,y) horizontal plane the velocity of the robot is defined by $v_1 = v_r \cos(\Phi_r)$ (from 3.5). The angle formed by the robot with the (x,y) plane is $\Psi_r$ as we know. The projection of the spherical obstacle on the horizontal plane is denoted by a circle named $O_{\text{bxy}}$ as shown in figure 8 (a). Now, the robot is moving towards the obstacle. At any given point in time (let us assume at time $t_1$), the robot encounters an obstacle within its line of sight. Let us say that at time $t_1$, the robot (assumed as a point) forms a cone with the circular obstacle in 2 dimensional (x,y) plane.
as shows in figure 8 (a). The angle formed by this cone (let us call it collision cone) is denoted by the term \( \epsilon_1 \). \( A_{xy} \) and \( B_{xy} \) are the waypoints in (x,y) plane towards which the robot deviates in order to avoid collision with the obstacle. The deviation in the horizontal plane is called “zero slope deviation” and is discussed in detail in section 4.1 [1].

Similarly, in the vertical (r,z) plane the velocity of the robot is denoted by \( v_r \). The angle formed by the robot with the (r,z) plane is \( \varphi_r \) as we know. The angle formed by the collision cone in the (r,z) plane is denoted by \( \epsilon_2 \). The projection of the obstacle in the vertical plane is denoted by \( O_{brz} \). The waypoints in the vertical plane are denoted by \( A_{rz} \) and \( B_{rz} \). The deviation of the robot towards the waypoints in the vertical plane is called “infinite slope deviation” which is discussed in detail in section 4.2 [1].

4.1 Zero slope deviation and infinite slope deviation

The deviation of the robot towards the waypoint in the horizontal (x,y) plane is called zero slope deviation. In this case, when the robot encounters an obstacle in the 3 dimensional space it maneuvers around it in the horizontal direction i.e. deviates to either the left or the right to bypass the obstacle. Zero slope deviation is achieved by changing the values of the navigation parameters \( N_1 \) and \( c_1 \) for the horizontal plane. Collision takes
place in the horizontal plane if the value of the heading angle $\Psi_r$ lies in the interval $[\sigma_{xy} - (\varepsilon_1)/2, \sigma_{xy} + (\varepsilon_1)/2]$. In order to avoid collision in this case, the desired value of the heading angle should not lie in the interval mentioned above. The new value of the desired heading angle can be given by the equation:

$$\Psi_{r,\text{des}} = N_{1\text{new}} \sigma_{xy,p} + c_{1\text{new}}$$

where,

$$\Psi_{r,\text{des}} = \text{desired value of heading angle to avoid collision}$$

$$N_{1\text{new}} = \text{new value of proportionality factor } N_1 \text{ for } (x,y) \text{ plane for which collision avoidance can be achieved successfully}$$

$$c_{1\text{new}} = \text{new value of deviation factor } c_1 \text{ for } (x,y) \text{ plane for which collision avoidance can be achieved successfully}$$

$$\sigma_{xy,p} = \text{line of sight angle for waypoint in } (x,y) \text{ plane}$$

The deviation of the robot towards the waypoint in the vertical $(r,z)$ plane is called infinite slope deviation. In this case, when the robot encounters an obstacle in the 3 dimensional space, it maneuvers around it in the vertical direction i.e. deviates up or down to bypass the obstacle. Infinite slope deviation is achieved by changing the values of the navigation parameters $N_2$ and $c_2$ for the vertical plane. Collision takes place in the vertical plane if the value of the flight path angle $\Phi_r$ lies in the interval $[\sigma_{rz} - (\varepsilon_2)/2, \sigma_{rz} + (\varepsilon_2)/2]$. In order to avoid collision in this case the desired value of the flight path angle
should not lie in the interval mentioned above. The new value of the desired flight path angle can be given by the equation:

\[
\theta_{r_{des}} = N_{2new} \sigma_{rz, p} + c_{2new}
\]  

(20)

where, \( \theta_{r_{des}} \) = desired value of flight path angle to avoid collision

\( N_{2new} \) = new value of proportionality factor N2 for (r,z) plane for which collision avoidance can be achieved successfully

\( c_{2new} \) = new value of deviation factor c2 for (r,z) plane for which collision avoidance can be achieved successfully

\( \sigma_{rz, p} \) = line of sight angle for waypoint in (r,z) plane

4.2 Derivation of the new navigation parameters for obstacle avoidance

The new values of navigation parameters should be chosen so as to not only avoid collision but also to maintain a smooth trajectory. This implies that the new values of the proportionality factors should be chosen so that they satisfy the dynamic constraints of the 3 dimensional space (from 3.5). Assuming that the vehicle starts deviating at time t1, the heading and flight path angles of the vehicle at time t1 can be defined as \( \Psi_{r_{t1}} \) and \( \theta_{r_{t1}} \) respectively. For successfully achieving obstacle avoidance and maintaining a smooth trajectory, the values of the flight path and the heading angles at time t1 should
advance smoothly to the desired values $\Psi_{r_{des}}$ and $\Phi_{r_{des}}$. This can be expressed by the following equations:

$$\Psi_{r_{t1}} = H_{new} \sigma_{xy} p(t1) + c_{1new} + b_{1new} \tag{21}$$

$$\Phi_{r_{t1}} = H_{new} \sigma_{rz} p(t1) + c_{2new} + b_{2new} \tag{22}$$

The algorithm for obstacle avoidance can now be developed by taking the values of $N_{1new}$ and $N_{2new}$ as 1, taking the values of $c_{1new}$ as $(\varepsilon_1)/2$ or $(-\varepsilon_1)/2$ and of $c_{2new}$ as $(\varepsilon_2)/2$ or $(-\varepsilon_2)/2$ and of $b_{1new}$ and $b_{2new}$ to satisfy smoothness of trajectory [1].

### 4.3 Examples of obstacle avoidance

Obstacle avoidance in the horizontal and vertical planes has been discussed in section 4.1. The practical application of zero slope deviation and infinite slope deviation is illustrated with examples in this section.

**Zero slope deviation:** It has been proved that deviation in the horizontal (x,y) plane is achieved by deviating to either the left or the right of the obstacle. This is achieved by changing the navigation parameters - proportionality factor $N_1$ and deviation factor $c_1$ for horizontal plane. As per figure 8 (a) it is determined that zero slope deviation can be attained by deviating towards the waypoints $A_{xy}$ or $B_{xy}$ in the horizontal plane. At a given time (say $t_1$) when the robot encounters an obstacle in the path it begins its
deviation by following an alternate path either towards $A_{xy}$ or $B_{xy}$. The condition for collision avoidance in the horizontal plane is that the heading angle $\Psi_r$ should not lie in the interval $[\sigma_{xy} - (\varepsilon_1)/2, \sigma_{xy} + (\varepsilon_1)/2]$, where $\sigma_{xy}$ is the line of sight angle in the horizontal plane and $\varepsilon_1$ is the collision cone width angle in the horizontal plane. This condition along with (19) provides the desired deviation and obstacle avoidance in the horizontal plane. Zero slope deviation can be displayed graphically as follows:

Figure 9 – Motion of robot in the horizontal plane in the presence of obstacles (a) without deviation – collision occurs (b) with zero slope deviation – collision avoided

Infinite slope deviation: It has been proved that deviation in the vertical $(r,z)$ plane is achieved by deviating either upward or downward of the obstacle. This is achieved by
changing the navigation parameters - proportionality factor $N_2$ and deviation factor $c_2$ for the vertical plane. As per figure 8 (b) it is determined that infinite slope deviation can be attained by deviating towards the waypoints $A_{rz}$ or $B_{rz}$ in the vertical plane. At a given time (say $t_1$) when the robot encounters an obstacle in the path, it begins its deviation by following an alternate path either towards $A_{rz}$ or $B_{rz}$. The condition for collision avoidance in the vertical plane is that the flight path angle $\phi_r$ should not lie in the interval $[\sigma_{rz} - (\varepsilon_2)/2, \sigma_{rz} + (\varepsilon_2)/2]$, where $\sigma_{rz}$ is the line of sight angle in the vertical plane and $\varepsilon_2$ is the collision cone width angle in the vertical plane. This condition along with (20) provides the desired deviation and obstacle avoidance in the vertical plane. Infinite slope deviation can be illustrated graphically as follows:

![Diagram](image)

Figure 10 – Motion of robot in the vertical plane in presence of obstacles (a) without deviation – collision occurs (b) with infinite slope deviation – collision avoided
5.1 Simulation results for surveillance mode

The initial position of the target is assumed as (10,5,3) and that of the robot is assumed as (12,6,3). The robot speed is assumed to be greater than the target speed.

Figure 11 – The initial position of the robot (red dot) and the target (green dot) at time t=0 in 3 dimensional space
Figure 12 – Initial position of robot (red star) and the target (green star located on extreme left bottom) in the horizontal (x,y) plane at time \( t=0 \)

Figure 13 – Initial position of the robot (red star) and the target (not visible at time \( t=0 \) due to the constraints provided) in the vertical (r,z) plane at time \( t=0 \)
Figure 14 – The position of the robot (red dot) and the target (green dot) after 5 steps in the 3 dimensional space (we can see that the robot is gradually approaching the target)

Figure 15 – The position of the robot (red star) and the target (green star) after 5 steps in the horizontal (x,y) plane
Figure 16 - The position of the robot (red star) and the target (green star) after 5 steps in the vertical (r,z) plane.

Figure 17 - The position of the robot (red dot) and the target (green dot) after 10 steps in the 3 dimensional space (we can see that the robot is even closer to the target).
Figure 18 – The position of the robot (red star) and the target (green star) after 10 steps in the horizontal (x,y) plane

Figure 19 - The position of the robot (red star) and the target (green star) after 10 steps in the vertical (r,z) plane
Figure 20 - The position of the robot (red dot) and the target (green dot) after 15 steps in the 3 dimensional space (we can see that the robot has now reached the target)

Figure 21 – The position of the robot (red star) and the target (green star) after 15 steps in the horizontal (x,y) plane
Figure 22 - The position of the robot (red star) and the target (green star) after 15 steps in the vertical (r,z) plane.

Figure 23 - The position of the robot (red dot) and the target (green dot) after 30 steps in the 3 dimensional space (we can see that the robot has once reached the target, bypassed it and is headed again towards it).
Figure 24 – The position of the robot (red star) and the target (green star) after 30 steps in the horizontal (x,y) plane

Figure 25 - The position of the robot (red star) and the target (green star) after 30 steps in the vertical (r,z) plane
Figure 26 - The position of the robot (red dot) and the target (green dot) after 35 steps in the 3 dimensional space (we can see that the robot has once reached the target, bypassed it and is reached it again)

Figure 27 – The position of the robot (red star) and the target (green star) after 35 steps in the horizontal (x,y) plane
Figure 28 - The position of the robot (red star) and the target (green star) after 35 steps in the vertical (r,z) plane
5.2 Simulation results for obstacle avoidance mode

It is assumed that the robot deviates in the horizontal plane only towards a waypoint A. The values assumed for input variables are \( c_1 = \pi/4, \ x_0 = 13, \ z_0 = 3, \ y_0 = 7, \ y_0 = 5, \ z_0 = 4, \ x_A = 3, \ y_A = 4. \)

![Figure 29 - The initial position of the robot (red dot) and the target (green dot) at t=0 in the 3 dimensional space](image)

![Figure 30 - The initial position of the robot (red star) and the target (green star located at top right corner) at time t=0 in the horizontal (x,y) plane](image)
Figure 31 – Initial position of the robot (red star) and target (not visible at time t=0 due to the constraints provided) at time t=0 in the vertical (r,z) plane

Figure 32 - The position of the robot (red dot) and the target (green dot) after 15 steps in the 3 dimensional space (we can see that the robot is advancing towards the moving target)
Figure 33 – The position of the robot (red star) and the target (green star) after 15 steps in the horizontal (x,y) plane

Figure 34 - The position of the robot (red star) and the target (green star) after 15 steps in the vertical (r,z) plane
Figure 35 - The position of the robot (red dot) and the target (green dot) after 22 steps in the 3 dimensional space (this is the time when the robot encounters an obstacle and begins deviation towards waypoint A)

Figure 36 - The initial position of the alternative path followed by the robot (red dot) and the target (green dot) on encountering an obstacle
Figure 37 – The position of the robot (red star) and the target (green star) after 22 steps in the horizontal (x,y) plane

Figure 38 - The position of the robot (red star) and the target (green star) after 22 steps in the vertical (r,z) plane
Figure 39 - The position of the robot (red dot) and the target (green dot) after 50 steps in the 3 dimensional space (we can clearly see that the robot has deviated from its original course and is approaching the target gradually by following the alternate path)

Figure 40 – The position of the alternative path followed by the robot (red dot) and the target (green dot) after 28 steps on encountering an obstacle (the robot appears to be gradually approaching the target)
Figure 41 – The position of the robot (red star) and the target (green star) after 50 steps in the horizontal (x,y) plane

Figure 42 - The position of the robot (red star) and the target (green star) after 50 steps in the vertical (r,z) plane
This project describes kinematics based linear navigation laws applied to reactive path planning and navigation for Unmanned Aerial Vehicles. Kinematics equations are defined for the motion of a UAV and the terms involved in those equations are systematically derived and explained. The plane decomposition method is introduced, which greatly simplifies the problem of path planning in a 3 dimensional space by dividing it into two 2 dimensional planes. The line of sight angles in each plane are then defined based on which the flight path angle and heading angle of the robot are determined. Control laws are then derived for the flight path and heading angles which take into account the navigation parameters including proportionality and deviation factors. These navigation parameters are used with varying degrees in order to attain satisfactory trajectory control, including path smoothness and obstacle avoidance. The obstacle avoidance mode is discussed in detail, where the collision cone concept is used. The concept of waypoints towards which the robot deviates in order to follow an alternate path is also described. Moreover, the dynamic constraints restricting the motion of the robot and the target in the 3 dimensional space are also taken into account. The concept of heading regulation for determining the final values of desired navigation angles is also discussed. Finally, MATLAB code is used to simulate the suggested methods in the two
cases: (1) Surveillance mode and (2) obstacle avoidance mode. Simulations show the effectiveness of the methods.
File name : drone.m (SURVEILLANCE MODE)

%input ('target speed:')</n

\[ v_G = 2; \]

%input ('robot speed:')</n

\[ v_r = 3; \]

% initial positions of the target

\[ x_{g0} = \text{input ('initial x position of G :')} \]

\[ y_{g0} = \text{input ('initial y position of G :')} \]

\[ z_{g0} = \text{input ('initial z position of G :')} \]

% initial positions of the robot

\[ x_{r0} = \text{input ('initial x position of R :')} \]

\[ y_{r0} = \text{input ('initial y position of R :')} \]

\[ z_{r0} = \text{input ('initial z position of R :')} \]

% proportionality factors

\[ N_1 = 2; \]

\[ N_2 = 2; \]

% step size of robot motion

\[ h = 0.1; \]
% final time when execution is halted

\( t_f = 20-h; \)

% for loop executing the main body of robot motion

\textbf{for} \( t = 0:h:t_f \)

% projection of euclidean distance of goal from origin on \((x, y)\) plane

\( r_1 = \sqrt{(y_g0)^2 + (x_g0)^2}; \)

% projection of line of sight from robot to goal on \((x, y)\) plane

\( r_\theta = \sqrt{(y_\theta0 - y_\theta)^2 + (x_\theta0 - x_\theta)^2}; \)

% line of sight angle in vertical plane for the goal

\( \sigma_{nv0} = \text{atan2}(z_g0, r_1); \)

% line of sight angle in horizontal plane for the goal

\( \sigma_{nh0} = \text{atan2}(y_g0, x_g0); \)

% heading angle for the goal

\( \psi_g = N_1 \cdot \sigma_{nh0}; \)

% flight path angle for the goal

\( \varphi_g = N_2 \cdot \sigma_{nv0}; \)

% line of sight angle in vertical plane for the robot

\( \sigma_{nv} = \text{atan2}(z_\theta0 - z_\theta, r_\theta); \)

% line of sight angle in horizontal plane for the robot

\( \sigma_{nh} = \text{atan2}(y_\theta0 - y_\theta, (x_\theta0 - x_\theta)); \)
% heading angle for the robot
\[ \psi = N_1 \cdot \sigma_{nh}; \]

% flight path angle for the robot
\[ \varphi = N_2 \cdot \sigma_{nv}; \]

% equations defining the motion of the target
\[ x_g = v_g \cdot \cos(\varphi_g) \cdot \cos(\psi_g) \cdot h + x_{g0} \]
\[ y_g = v_g \cdot \cos(\varphi_g) \cdot \sin(\psi_g) \cdot h + y_{g0} \]
\[ z_g = v_g \cdot \sin(\varphi_g) \cdot h + z_{g0} \]

% equations defining the motion of the robot
\[ x_r = v_r \cdot \cos(\varphi) \cdot \cos(\psi) \cdot h + x_0 \]
\[ y_r = v_r \cdot \cos(\varphi) \cdot \sin(\psi) \cdot h + y_0 \]
\[ z_r = v_r \cdot \sin(\varphi) \cdot h + z_0 \]

% curve displaying the motion of the target in horizontal plane
figure(808)
plot({x}_g0, {y}_g0, 'g*')
hold on
% curve displaying the motion of the robot in horizontal plane
plot({x}_r, {y}_r, 'r*')
xlabel('x')
figure(8008)
% curve displaying the motion of the target in the vertical plane
plot($r_0, z_0, 'g*$')
hold on
% curve displaying the motion of the robot in the vertical plane
plot($r_0, z_r, 'r*$')
xlabel('r')
figure(020)
% curve displaying the motion of the robot in 3 dimensional space
plot3($x_r, y_r, z_r, '.r$')
hold on
% curve displaying the motion of the target in 3 dimensional space
plot3($x_g, y_g, z_g, '.g$')
hold on
% derivation of new values of target position from old values based
% on for loop
$x_{g0} = x_g$
$y_{g0} = y_g$
$z_{g0} = z_g$
% derivation of new values of robot position from old values based
% on for loop
$x_0 = x_r$
$y_0 = y_r$
$z_0 = z_r$
pause
end

File name: obstacle1.m (OBSTACLE AVOIDANCE MODE)

%input ('target speed:')</n
v_g = 3;

%input ('robot speed:')</n
v_r = 5;

c_1 = input('initial deviation angle in xy plane')</n
x_g0 = input('initial x position of G:')</n
y_g0 = input('initial y position of G:')</n
z_g0 = input('initial z position of G:')</n
x_0 = input('initial x position of R:')</n
y_0 = input('initial y position of R:')</n
z_0 = input('initial z position of R:')</n
x_A = input('x position of waypoint A:')</n
y_A = input('y position of waypoint A:')</n
N_1 = 3; % proportionality factor in (x,y) plane

N_2 = 3; % proportionality factor in (r,z) plane
$N_{1_{new}} = 5$; % new value of proportionality factor for obstacle avoidance

coll = 0; % initially collision is assumed to not take place

h = 0.1; % step size

$t_f = 10 - h$; % final time after which execution stops

for t = 0:h:$t_f$ % for loop executing the body of code

r = 3; % radius of the spherical obstacle

m = 5; % linear distance in xy plane between obstacle and robot

$\varepsilon = \text{atan2}(r,m)$ % collision cone width angle

% projection of euclidean distance from origin to goal on (x,y) plane

$r_1 = \sqrt{((y_0)^2 + (x_0)^2)}$;

% projection of line of sight from robot to goal on (x,y) plane

$r_0 = \sqrt{((y_0 - y_0)^2 + (x_0 - x_0)^2)}$;

% line of sight angle in vertical plane for the goal

$\sigma_{nv0} = \text{atan2}(z_0, r_1)$;

% line of sight angle in horizontal plane for the goal

$\sigma_{nh0} = \text{atan2}(y_0, x_0)$;

% heading angle for the goal

$\psi_g = N_1*\sigma_{nh0}$;

% flight path angle for the goal

$\phi_g = N_2*\sigma_{nv0}$;
% line of sight angle in vertical plane for the robot
\[ \sigma_{nv} = \text{atan2}((z_{g0} - z_0), r_0); \]

% line of sight angle in horizontal plane for the robot
\[ \sigma_{nh} = \text{atan2}((y_{g0} - y_0), (x_{g0} - x_0)); \]

% heading angle for the robot
\[ \psi = N_1 \cdot \sigma_{nh}; \]

% flight path angle for the robot
\[ \varphi = N_2 \cdot \sigma_{nv}; \]

% equations defining the motion of the target
\[ x_g = v_g \cdot \cos(\varphi_g) \cdot \cos(\psi_g) \cdot h + x_{g0} \]
\[ y_g = v_g \cdot \cos(\varphi_g) \cdot \sin(\psi_g) \cdot h + y_{g0} \]
\[ z_g = v_g \cdot \sin(\varphi_g) \cdot h + z_{g0} \]

% equations defining the motion of the robot
\[ x_r = v_r \cdot \cos(\varphi_r) \cdot \cos(\psi_r) \cdot h + x_0 \]
\[ y_r = v_r \cdot \cos(\varphi_r) \cdot \sin(\psi_r) \cdot h + y_0 \]
\[ z_r = v_r \cdot \sin(\varphi_r) \cdot h + z_0 \]

% collision detection formula
if \[ \psi = \pm \psi_{\text{NH}} \pm \epsilon \]
\[ \text{coll} = 1 \]
end

% alternate path definition in case of collision
if coll == 1

% line of sight angle of the waypoint

\sigma_{\text{nA}} = \text{obsmodh}(y_g0, y_A, x_g0, x_A)

% new value of heading angle of robot

\psi_{\text{new}} = N_{1\text{new}} \times \sigma_{\text{nA}} + c1;

% new equations defining the motion of the goal

x_g = v_g \times \cos(\varphi_g) \times \cos(\psi_g) \times h + x_g0

y_g = v_g \times \cos(\varphi_g) \times \sin(\psi_g) \times h + y_g0

z_g = v_g \times \sin(\varphi_g) \times h + z_g0

% new equations defining the motion of the robot

x_r = v_r \times \cos(\varphi) \times \cos(\psi_{\text{new}}) \times h + x_0

y_r = v_r \times \cos(\varphi) \times \sin(\psi_{\text{new}}) \times h + y_0

z_r = v_r \times \sin(\varphi) \times h + z_0

% curve displaying the alternate path

figure (7007)

plot3(x_r, y_r, z_r, '.r')

hold on

plot3(x_g, y_g, z_g, '.g')

hold on

x_{g0} = x_g

y_{g0} = y_g
\[ z_{g0} = z_g \]
\[ x_0 = x_r \]
\[ y_0 = y_r \]
\[ z_0 = z_r \]

pause

tex
\end

% curve displaying the motion of robot and target in horizontal
% plane
figure(808)
plot(\(x_{g0}, y_{g0}, 'g*'\))
hold on
plot(\(x_r, y_r, 'r*'\))
xlabel('x')

% curve displaying the motion of the robot and target in
% vertical plane
figure(8008)
plot(\(r_0, z_{g0}, 'g*'\))
hold on
plot(\(r_0, z_r, 'r*'\))
xlabel('r')

% curve displaying the motion of the robot and target in 3
% dimensional space
figure(020)

plot3(x_r, y_r, z_r, '.r')
hold on

plot3(x_g, y_g, z_g, '.g')
hold on

x_g0 = x_g
y_g0 = y_g
z_g0 = z_g
x_0 = x_r
y_0 = y_r
z_0 = z_r

pause
end

% function defining the line of sight angle of the waypoint

function \( \sigma_{nhC} = \text{obsmodh}(y_g0, y_c, x_g0, x_c) \)

% calculation of line of sight angle of waypoint

\( \sigma_{nhC} = \arctan2((y_g0 - y_c), (x_g0 - x_c)) \)
end
REFERENCES


http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5723778&contentTyp e=Journals+%26+Magazines&searchField%3DSearch_All%26queryText%3DReacti ve+Path+Planning+for+3-D+Autonomous+Vehicles


http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1430279&contentTyp e=Conference+Publications&searchField%3DSearch_All%26queryText%3DVision-Based+Control+of+Micro%E2%80%93Air%E2%80%93Vehicles%3A+Progress+and+Problems+In+Estimation


[4]. Hierarchical Control of Small Autonomous Helicopters. Christopher P. Sanders, Paul A. DeBitetto, Eric Feron, Hon Fai Vuong and Nancy Leveson.

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=761744&contentType=Conference+Publications&searchField=Search_All&queryText=Hierarchical+Control+of+Small+Autonomous+Helicopters


http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1545112&contentType=Conference+Publications&searchField=Search_All&queryText=ROBUST+FEEDBACK+LINEARIZATION+AND+GH1+CONTROLLER+FOR+A+QUADROTOR+UNMANNED+AERIAL+VEHICLE