Chapter 5

Waypoints Generation

The waypoints or trajectory following logic is modeled inside the Path Planning subsystem of Flight Control System of the Minidrone Model, which is a part of the UAV Toolbox of the MATLAB, as shown in Figure 5.1

In this thesis, the waypoint follower subsystem is used as following:

- *Follow a Trajectory:* Using [x y z] coordinates, define a set of waypoints for the minidrone to follow. We employ thirteen waypoints in this project, the first coordinate denoting the UAV's initial position,the last coordinate defining the UAV's final position, and the remaining points defining UAV's trajectory.
- *Lookahead Distance:* Define the UAV's lookahead distance as it navigates along the path. The lookahead distance in this project is 0.25 meters. This value is proportional to the speed at which the UAV will fly (as the lookahead distance is increased, the drone will fly faster along the waypoints).
- *Update the Status:* Once the UAV's navigation along the square path is complete, update the status. In this project, the Landing Logic subsystem receives the navigation complete state (received from the Status port of this subsystem) and uses it to land the drone.

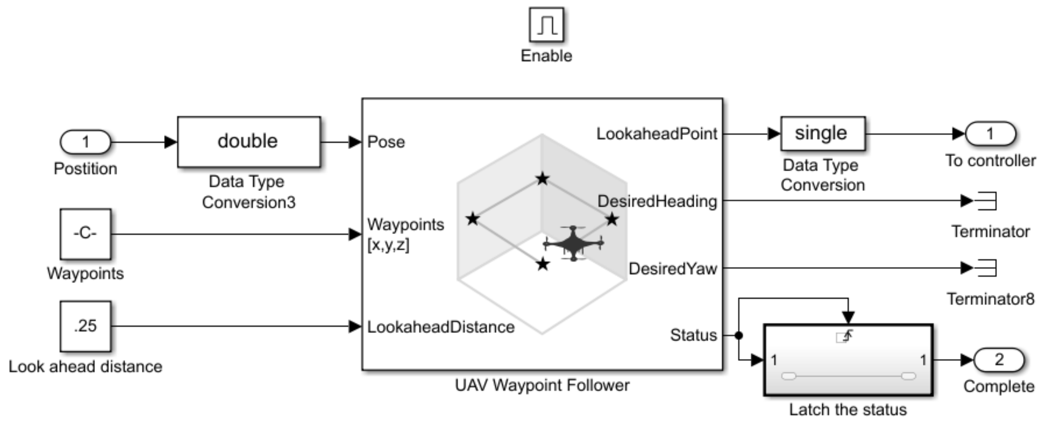

Figure 5.1: Waypoint Follower Subsystem

Waypoint List		
Standard Shapes	**Randomly Generated Waypoints**	**Gaussian Generated Waypoints**
Square	Random(1,2)	SD 1
Circular	Random(1,3)	SD 3
Pentagon	Random(1,3)	SD 3
Hexagon	Random(1,3)	SD 4
Total:4	Total:12	Total 15

Table 5.1: List of Generated Waypoints

The **waypoints** are sent to the subsystem using a parameterized block connected with the subsystems. This parameterized block gets initialized with the set of waypoints as a list or coordinates like following:

[0 0 -1; 1.5 0 -1; 1.5 1.5 -1; 0 1.5 -1; 0 0 -1; 1.5 0 -1; 1.5 1.5 -1; 0 1.5 -1; 0 0 -1].

Here the value of z axis denotes the height from the ground. To ensure that the UAV navigates on a plane, the value is kept at -1 (representing the height of 1 meter from the ground).

Following the aforementioned rules, a total 31 waypoints are created, considering various scenarios and also random conditions. These waypoints are shown in Table 5.1

As shown in the table, total 4 waypoints are generated maintaining standard shapes, i.e. square, circular, pentagon and hexagon. To allow the UAV to cover more scenarios, a total number of 12 trajectories are created by generating random numbers betwwen two single digit integers. Also 15 trajectories are generated following normally generated numbers of mean and standard deviation.

Chapter 6

Data Generation

In the previous chapter, all the theoretical background was discussed. In this chapter, a detailed process for data generation from the Simulink based UAV model is discussed. The first sub-chapter details all the considered features and the data collection strategies for fault-free case. Then the later sub-chapter explains all the considered faulty cases and data generation process for each of the cases.

6.1 Features Collected

As the goal of this Thesis is to detect and classify various sensor-level faults, all the sensor data from IMU sensor like, 3-axis Accelerometer, 3-axis Gyroscope, Altitude sensor data is collected during data collection. Besides, data from the Controller unit of the UAV and also Actuator level data are collected. At the end, data from a total number of 18 features are considered while data collection. These features are enlisted in Table 6.1. Although Data from the image sensor of UAV also plays a very important role in the UAV operation, this sensor is not considered in this thesis due to lack of suitable image fault injection mechanism.

6.2 Data Generation for Fault-free Case

Based on the generated 31 waypoints, discussed in Chapter 5, fault free data is collected from the UAV simulation model through extensive monitoring of the data quality. For

Feature Name	**Source**	**Number of Features**
3-axis Accelerometer	Sensor	3
3-axis Gyroscope	Sensor	3
Altitude	Sensor	1
Motor Command	Actuator	4
Roll-Pitch-Yaw	Actuator	3
Thrust ref out	Actuator	4
Total Number of Features		**18**

Table 6.1: List of Collected Features

convenience, the whole data generation process is automated. Figure 6.1 shows the whole flow-diagram for Fault-free data generation process.

After initializing the set of waypoints, and compiling the model, for each waypoint, the simulation is performed for 7 times,by increasing the noise seed by 5% everytime (0% -30%). This is done considering the real time scenario, where the noise seed is quite dynamic in nature, and also has a impact on the sensor's reading of the UAV, hence impact on the final state of the UAV as well.

So, considering the inner loop (for increasing noise seed) and also outer loop (for all the waypoints), the total number of 217 simulations are performed. Duration of each simulation is 60 Seconds.All the features 6.1 are converted to timeseries data type, and resampled to 0.01 Seconds for data compression.Finally all the simulated data are saved as CSV files.

6.3 Data Generation for Faulty Case

As discussed in Chapter 5, many different faults could occur in UAV's sensor and actuator unit.In this thesis, we consider only the sensor-level faults. To demonstrate fault classification mechanism, in this thesis, both Accelerometer and Gyroscope sensor are chosen, and faults are injected in these two sensors.In this thesis, total 4 faulty scenarios are considered. These are:

1. Bias/Offset Fault,
2. Stuck-at Fault,
3. Noise Fault, and
4. Package-drop Fault

In the following part of this sub-chapter, all the processes of data collection for these faults are discussed.Although the data collection strategies are implemented in simulation scenario,during faulty data generation, we considered the following assumptions keeping in mind about the real world case.

1. Fault could be introduced at any point of time during the simulation.
2. Fault could last for any duration of time during the simulation.
3. Fault value could be any value in case of Bias/Offset fault.

Simulink's Fault Injection Block (FI Block) is used for injecting these faults. For this purpose, Fault injection mechanism is integrated to the UAV model's sensor unit.Figure 6.2 and Figure 6.3 shows the integration of fault injection submodule in case of Accelerometer.The similar extension is done as well for the gyroscope.

FI block has a constant Flag generator, denoting when to inject the fault. In this thesis, we replaced this with non-uniform pulse generators, keeping in mind that fault

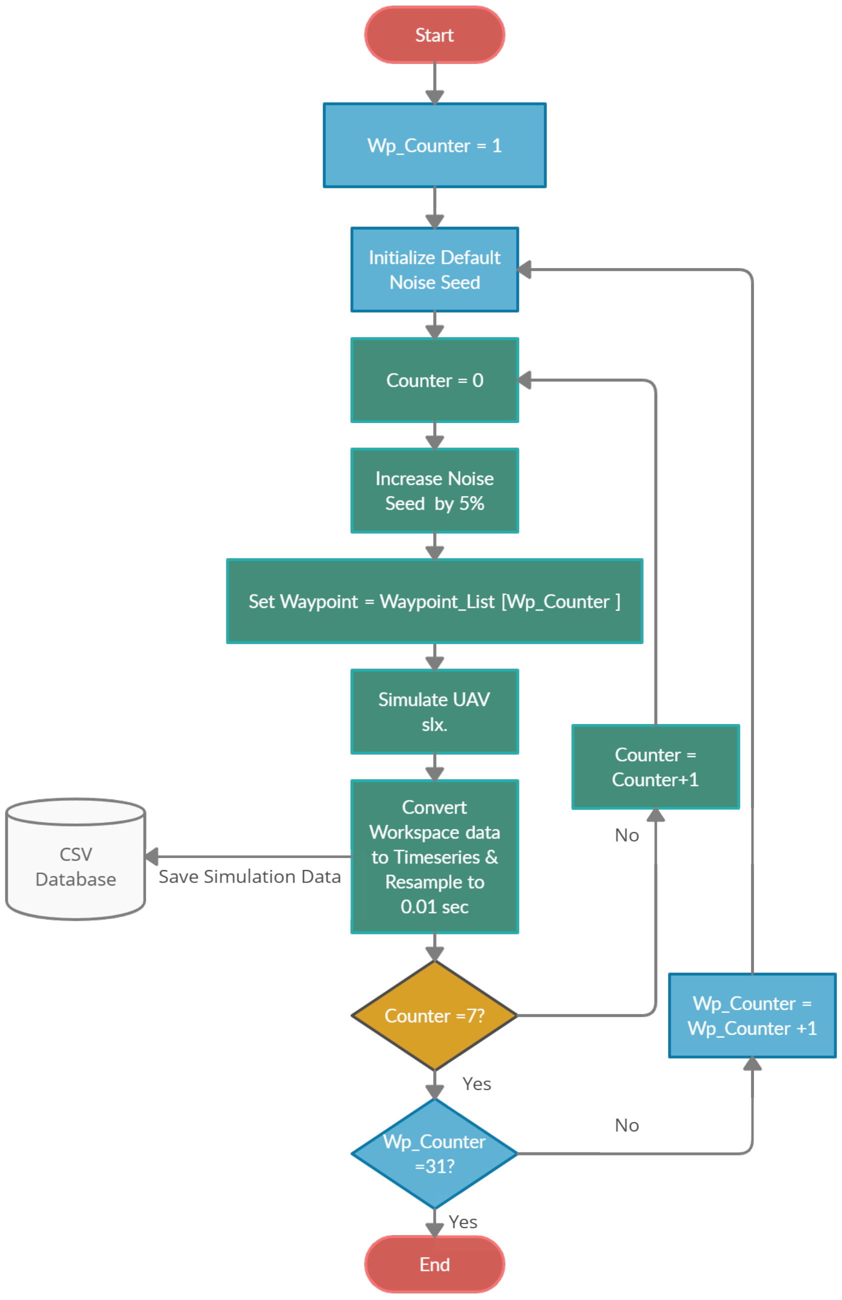
Start
Wp_Counter = 1
Initialize Default Noise Seed
Counter = 0
Increase Noise Seed by 5%
Set Waypoint = Waypoint_List [Wp_Counter]
Simulate UAV slx.
Convert Workspace data to Timeseries & Resample to 0.01 sec
Save Simulation Data
CSV Database
Counter = Counter+1
No
Counter =7?
Yes
Wp_Counter = Wp_Counter +1
No
Wp_Counter =31?
Yes
End

Figure 6.1: Fault Free Data Collection Process

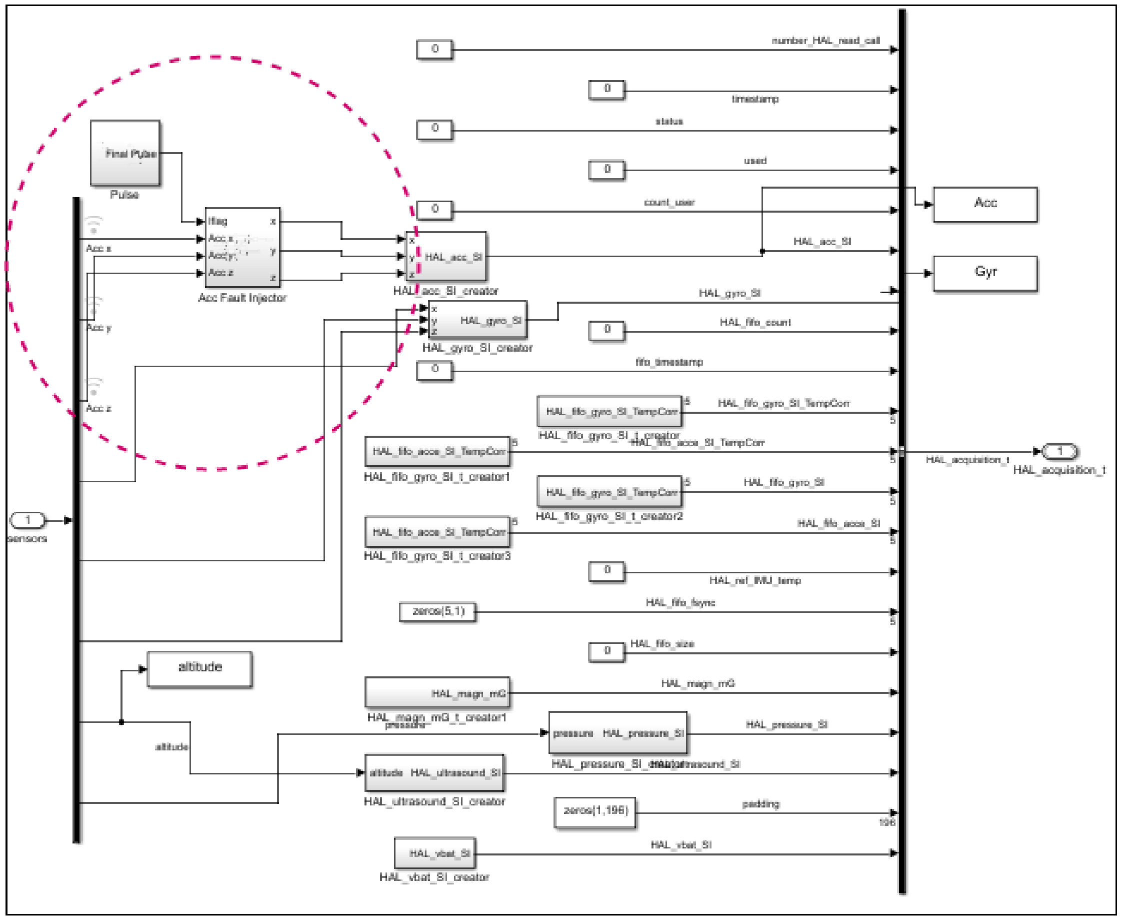

Figure 6.2: Integration of Fault Injection to Sensor Module for Accelerometer

could be introduced at any point of time. Hence, during the simulation, these non-uniform pulses can be randomly chosen and fault is then injected based on the pulse value (high or low, where fault can be injected when the pulse value is high). These pulse generators are shown in Figure 6.4

Initial time delay and duration of fault occurrence are set randomly, from a predefined range of values, to mimic the real time scenario. All the chosen values are listed in Table 6.2

In case of fault value, for stuck-at fault, the fault value is stuck to the value of last time instant. For Package-drop, the fault value is represented as a unique value of 15 (for understanding - as 15 is not observed anywhere else). Due to the constant velocity of the UAV, the acceleration is almost always close to zero, hence a small percentage of Noise injection is not suitable in this thesis. So, 100% noise is injected in both Accelerometer and Gyroscope during noise fault injection.Considering the real world case, bias/offset fault could be of any value. In this thesis, the values for bias/offset fault is limited to a range of values (Table 6.2)

Considering the aforementioned conditions, assumptions and fault nature, all the faults, i.e. bias/offset, stuck-at, noise, and package-drop fault are sequentially injected

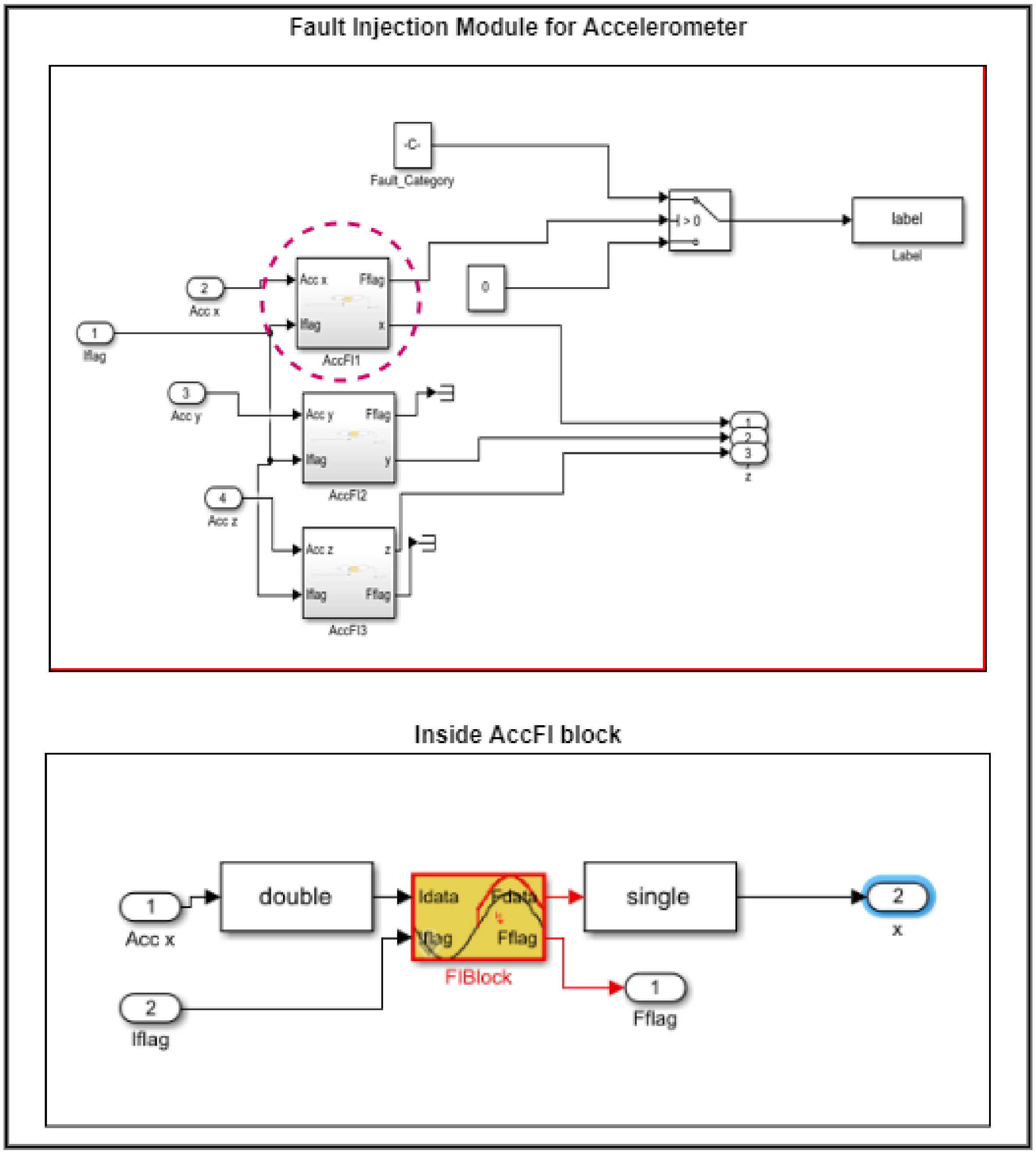

Figure 6.3: Fault Injection Submodule for Accelerometer

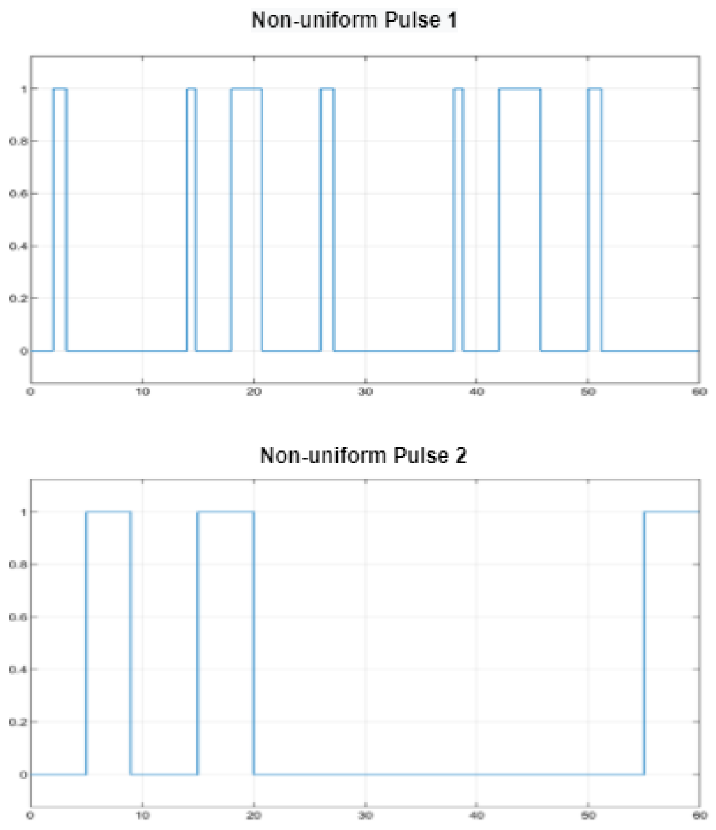

Figure 6.4: Non-uniform Pulses

Fault Category	**Fault Value**	**Fault Delay**	**Fault Duration**
Bias/Offset	2,3,4,5,6,7	0, 1, 2,3,4	0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5
Stuck-at	Stuck to last value	0, 1, 2,3,4	0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5
Noise	100%	0, 1, 2,3,4	0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5
Package Drop	15	0, 1, 2,3,4	0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5

Table 6.2: Values, delay and duration of different faults

Fault Category	**Faulty Accelerometer**	**Faulty Gyroscope**
Bias/Offset	114	114
Stuck-at	114	114
Noise	114	114
Package Drop	114	114

Table 6.3: Number of simulation during faulty data collection

in both Accelerometer (x axis) and Gyroscope (x axis) during the UAV simulation, using FI block []. The whole fault injection process is shown in the flow-diagram in Figure 6.5

The faulty data collection process is simulated for all 31 generated waypoints, but this time noise seed is kept at the default value. Instead an inner counter is used to consider many combinations of fault value, delay and fault duration, because for each simulation, these values are randomly chosen as per table 6.2.Thus for each waypoint, simulation is performed for a duration of 60 secs, for 4 times for each fault category and sensor (Accelerometer / Gyroscope). Similarly as fault-free data collection, the collected data is then converted to time-series and then resampled to 0.01 second. Finally the data is saved as a CSV file.

Hence, at the end of the simulation total 114 CSV files are generated for each fault category and sensor type. Table 6.3 shows total amount of files generated during data collection process, where every files contain total 6000 time series entries.

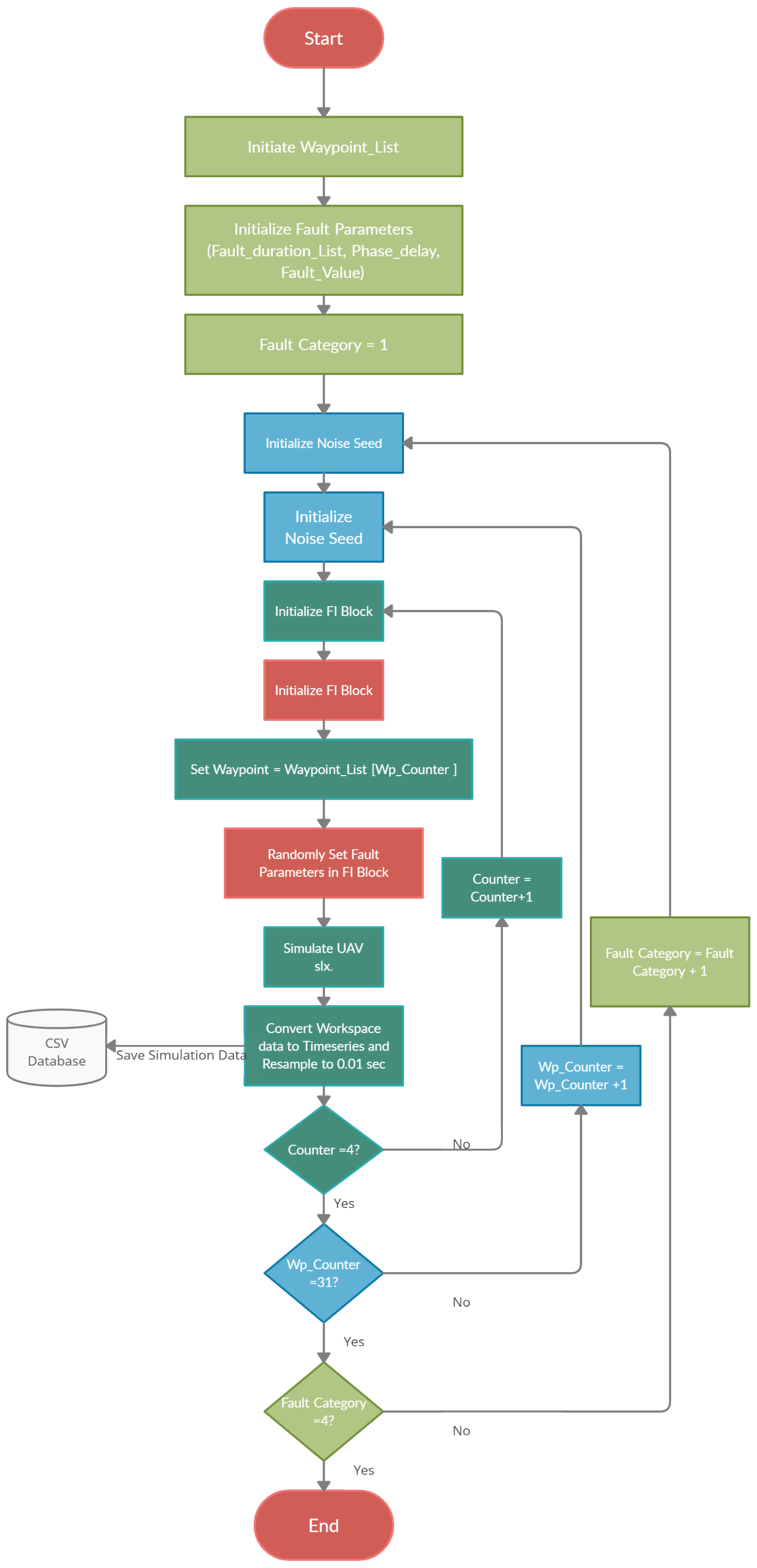
Start
Initiate Waypoint_List
Initialize Fault Parameters (Fault_duration_List, Phase_delay, Fault_Value)
Fault Category = 1
Initialize Noise Seed
Initialize Noise Seed
Initialize FI Block
Initialize FI Block
Set Waypoint = Waypoint_List [Wp_Counter]
Randomly Set Fault Parameters in FI Block
Simulate UAV slx.
Convert Workspace data to Timeseries and Resample to 0.01 sec
Save Simulation Data
CSV Database
Counter =4?
No
Counter = Counter+1
Yes
Wp_Counter =31?
No
Wp_Counter = Wp_Counter +1
Yes
Fault Category =4?
No
Fault Category = Fault Category + 1
Yes
End

Figure 6.5: Faulty Data Collection Process