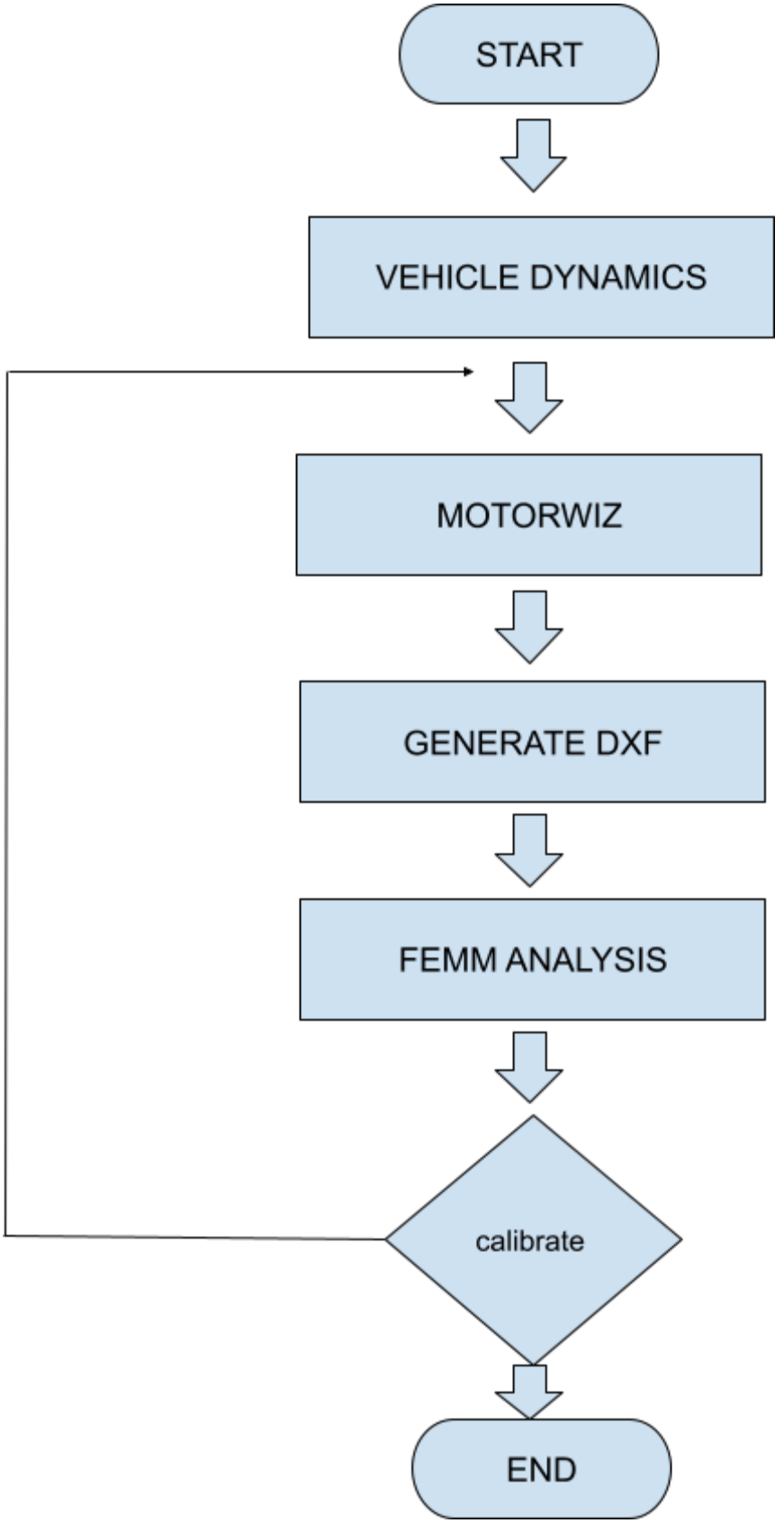
Motor Design Tool

Motor Design Tool:

The Motor Design Tool (MDT) is a software tool that allows its users to design and simulate electric motors from an end use case (Electric motor for an Electric Vehicle) or application.

From specifying the vehicle parameters and the vehicle performance specification, we can derive the specifications that define the Electric Motor. We can then proceed to design a motor and test the motor using analytical solutions as well as a Finite Element Method tool called FEMM (Finite Element Method Magnetics). The tool is intended to use in an iterative manner, in order to refine the design to the user's satisfaction.


START
VEHICLE DYNAMICS
MOTORWIZ
GENERATE DXF
FEMM ANALYSIS
calibrate
END

Project Structure:

Motor Design Tool:

Description: Repository that contains backend and frontend code base.

1. **Frontend:**
   Description: Code base that contains all the relevant code for the frontend. The frontend is a simple website created on Flutter Web Stack which is hosted on AWS S2 bucket and uses AWS CloudFront.

2. **server_multi_db.py:**
   Description: Python code that acts as the REST API endpoint between the frontend code and the backend code which is hosted on a server. It uses multiprocessing to run simulation as a background task whose results can be polled from the front end. It saves all the data in a mongodb database.

3. **Backend:**

   Description: Python code base that contains all the relevant code for the backend.

   1. **Vehicle Dynamics:**

      Description: Contains all the relevant code to derive vehicle requirements from vehicle specs. The output of this module are: continuous mode and continuous plot, peak mode and peak plot and voltage specs.

      1. **vehicle_dynamics.py**: contains **VehicleDynamics** class that takes vehicle parameters and performance specs from the user and generates the above mentioned outputs.

      2. **vehicle_dynamics_methods.py**: contains all the relevant functions used by the **VehicleDynamics** class.

2. Motor Wiz:

Description: Contains all the relevant code to derive motor geometry, winding pattern, motor parameters, motor cost and motor operating points from user input as well as from vehicle requirements derived from vehicle dynamics. The output of this module are: DXF of motor, DXF of winding, Internal calculated variables, performance specs and cost and weight of materials.

1. **Materials:**

   Description: Folder contains all the relevant configuration files and code to store and load material data. More materials can be added by modifying the configuration files.

   1. **magnet.ini**: Configuration file that contains various parameters that describe a particular magnet type

   2. **wire.ini**: Configuration file that contains various parameters of a wire of a particular type

   3. Steel.ini: Configuration file that contains various parameters of steel of a particular type.

   4. htc.ini : Configuration file that contains various parameters that define the thermal properties of various materials.

   5. load_materials.py: loads materials (steel,wire, magnet) based on user input.

   6. thermal.py: loads the thermal data of various materials.

2. Topology:

Description: Folder contains code that performs calculations in order to determine the geometry, performance, cost and weight of particular motor topology. (Right Now we are only using an IPMSM and SPMSM Radial topology)

1. **inner_rotor.py:**
   Performs the necessary calculations for inner rotor topology for the SPMSM and IPMSM motors.

3. motorwiz.py:

Based on data from user input, this creates a MotorWiz class in order to abstract and encapsulate data. This module also calls the above mentioned modules in order to generate relevant outputs.

3. Test Design:

Description: Contains all the relevant code to design a motor from the geometry obtained from motorwiz or user data and convert it into a DXF file on which FEMM analysis can be performed.

1. dxf.py:

Based on motor geometry input, either from the user or from the geometry created by motorwiz, this draws a cross section of the motor (IPMSM/SPMSM Radial) and also generates a winding diagram. Both the motor and the winding are then exported into DXF format. Based on topology it chooses which motor is to be drawn.

2. **run_femm.py:**

   Uses FEMM, library in order to import the above generated DXF of the motor. This module also adds all the material properties, circuits, points and contours in order to run simulations. The simulation results in:
   i) air gap flux plot,
   ii) air gap flux value Bg_avg,
   iii) magnetic flux density plot.
   iv) Ld, Lq, psi_m

   Based on topology it chooses which motor is to be simulated.

3. **Topology:**

   1. **DXF: Module contains the various topology of motors to be drawn**

      1. **IPMSM_dxf.py:** draws IPMSM motor
      2. **SPMSM_dxf.py:** draws SPMSM motor

   2. **FEMM: Module contains the various topology of motors to be simulated**

      1. **IPMSM_femm.py:** simulates IPMSM motor
      2. **SPMSM_femm.py:** simulates SPMSM motor