Calculation of Fluid Inertia and Added Mass Loads on an MHK Turbine in AeroDyn

Hannah Ross

**GENERAL CONSIDERATIONS**

This implementation plan details changes to OpenFAST to calculate fluid inertia and added mass loads on the blades and tower of a marine hydrokinetic (MHK) turbine. Fluid inertia loads are caused by fluid accelerations and added mass loads by body accelerations. Changes will be made primarily in the AeroDyn module. Inertia and added mass loads will be calculated in the AD\_CalcOutput subroutine and added to existing loads.

**Activating Features**

Currently, a separate flag in the AeroDyn primary input file can be set to “True” or “False” to activate or deactivate buoyancy. An “MHK” flag is also available in the glue code and AeroDyn driver input files to denote no MHK turbine (MHK = 0), a fixed MHK turbine (MHK = 1), or a floating MHK turbine (MHK = 2). Rather than adding separate flags for each MHK feature, the Buoyancy flag will be removed, and all MHK features will automatically be activated for fixed and floating MHK turbines. Users can turn off these individual features by setting the relevant coefficients to zero. The cavitation check flag will not be removed, and this feature can be activated or deactivated independently of the others.

**Blades and Tower Background and Assumptions**

Per-unit-length loads are estimated at each blade or tower node by calculating the inertia and added mass forces according to the appropriate terms from Morison’s equation. The resulting loads are summed with the previously calculated hydrodynamic and/or buoyant per-unit-length loads. Loads for the blades are applied at the aerodynamic center. Loads for the tower are applied at the centerline. Marine growth and end effects are neglected, and members are not allowed to cross the free surface (i.e., members are always fully submerged). Ballast is not considered. Nodes do not need to be uniformly spaced, and axial loads are neglected. The tower is assumed to be axisymmetric (with the same coefficients used in both transverse directions), but the blade is not (with different coefficients normal and tangential to the chord, as well as an added mass coefficient for pitch). The reference cross-sectional area for the blade normal and tangential terms is chord\*thickness (). This is expressed as , where .

**Element Orientation**

The instantaneous global orientation of the member at each node must be known to determine the magnitude of fluid and body accelerations acting normal () and tangential () to the chord for the blades and in the transverse directions for the tower. The rotational acceleration of the blades about the local axis must also be known for added mass loads. These values are taken from the Orientation field of the motion input meshes to AeroDyn, which are updated at every time step.

**Coordinate Systems**

For land-based wind turbines, at ground level. For offshore wind turbines, at the mean sea level (MSL), with positive pointing upwards. Fixed and floating MHK turbines use the offshore wind definition. However, AeroDyn and ElastoDyn inputs for fixed MHK turbines that refer to elevation are given relative to the seabed and adjusted internally. All InflowWind inputs for both fixed and floating inputs are given relative to the seabed.

**INITIALIZE**

The following changes will be made to the initialization routines to add and validate inputs, calculate parameters, and define additional outputs.

**Inputs**

* Add an input variable “BlCpn” (, normal to chord dynamic pressure coefficient) defined at each blade node to the AeroDyn blade input file
* Add an input variable “BlCan” (, normal to chord added mass coefficient) defined at each blade node to the AeroDyn blade input file
* Add an input variable “BlCpt” (, tangential to chord dynamic pressure coefficient) defined at each blade node to the AeroDyn blade input file
* Add an input variable “BlCat” (, tangential to chord added mass coefficient) defined at each blade node to the AeroDyn blade input file
* Add an input variable “BlCam” (, pitch added mass coefficient) defined at each blade node to the AeroDyn blade input file
* Add an input variable “t\_c” (, thickness-to-chord ratio) defined at each blade node to the AeroDyn blade input file
* Add an input variable “TwrCp” (, transverse dynamic pressure coefficient) defined at each tower node to the AeroDyn primary input file
* Add an input variable “TwrCa” (, transverse added mass coefficient) defined at each tower node to the AeroDyn primary input file

**Validation**

* The added mass coefficients “BlCan”, “BlCat”, “BlCam”, and “TwrCa” should not be less than zero.
* The thickness to chord ratio “t\_c” should not be less than zero.

**Parameters**

The calculation of inertia and added mass loads includes some parameters that do not change with time. These parameters will be calculated during initialization and passed to relevant subroutines rather than recalculated at each time step.

Blade Parameters

is the fluid density

is the blade chord length at node

is the blade thickness-to-chord ratio at node

is the normal to chord inertia factor at blade node

is the tangential to chord inertia factor at blade node

is the normal to chord added mass factor at blade node

is the tangential to chord added mass factor at blade node

is the pitch added mass factor at blade node

Tower Parameters

is the diameter of tower node (i.e., tower characteristic length)

is the transverse inertia factor at tower node

is the transverse added mass factor at tower node

**Outputs**

Inertia and added mass loads will be calculated by AeroDyn in the global coordinate system and added to the existing loads that are passed to other modules. Additionally, the following user-selectable outputs will be added or modified.

New outputs (inertia and added mass loads)

* Inertia loads per unit length at each tower node, in the tower coordinate system
  + TwN#Fmx (x-component of inertia force per unit length at tower node)
  + TwN#Fmy (y-component of inertia force per unit length at tower node)
* Inertia loads per unit length at each blade node, in the blade coordinate system
  + B#N#Fmn (inertia force normal to chord per unit length at blade node)
  + B#N#Fmt (inertia force tangential to chord per unit length at blade node)
* Added mass loads per unit length at each tower node, in the tower coordinate system
  + TwN#Fax (x-component of added mass force per unit length at tower node)
  + TwN#Fay (y-component of added mass force per unit length at tower node)
* Added mass loads per unit length at each blade node, in the blade coordinate system
  + B#N#Fan (added mass force normal to chord per unit length at blade node)
  + B#N#Fat (added mass force tangential to chord per unit length at blade node)
  + B#N#Mam (added mass pitching moment per unit length at blade node)

Modified outputs (inertia and added mass loads added to existing rotor fluid loads)

* Total aerodynamic, buoyant, inertia, and added mass loads integrated over the rotor (including blade and hub loads), in the hub coordinate system
  + RtAeroFxh (total rotor aerodynamic, buoyant, inertia, and added mass force in x direction)
  + RtAeroFyh (total rotor aerodynamic, buoyant, inertia, and added mass force in y direction)
  + RtAeroFzh (total rotor aerodynamic, buoyant, inertia, and added mass force in z direction)
  + RtAeroMxh (total rotor aerodynamic, buoyant, inertia, and added mass moment in x direction)
  + RtAeroMyh (total rotor aerodynamic, buoyant, inertia, and added mass moment in y direction)
  + RtAeroMzh (total rotor aerodynamic, buoyant, inertia, and added mass moment in z direction)
* Total aerodynamic, buoyant, inertia, and added mass loads integrated over the rotor (including blade and hub loads), in the global coordinate system
  + RtAeroFxg (total rotor aerodynamic, buoyant, inertia, and added mass force in x direction)
  + RtAeroFyg (total rotor aerodynamic, buoyant, inertia, and added mass force in y direction)
  + RtAeroFzg (total rotor aerodynamic, buoyant, inertia, and added mass force in z direction)
  + RtAeroMxg (total rotor aerodynamic, buoyant, inertia, and added mass moment in x direction)
  + RtAeroMyg (total rotor aerodynamic, buoyant, inertia, and added mass moment in y direction)
  + RtAeroMzg (total rotor aerodynamic, buoyant, inertia, and added mass moment in z direction)
* Inflow accelerations?

**CALCULATE OUTPUT**

The following section details the fluid inertia and added mass calculations for the blades and tower. Forces will be calculated inside the AD\_CalcOutput subroutine after hydrodynamic and/or buoyancy calculations have been completed. Inertia and added mass loads will then be summed with the hydrodynamic and/or buoyant loads before they are passed to other modules.

**Blades**

* Calculate per-unit-length inertia and added mass forces in local blade coordinates at each node according to the appropriate terms from Morison’s equation
* Calculate per-unit-length added mass pitching moment in local blade coordinates each node according to the appropriate terms from Morison’s equation
* Convert loads to the global coordinate system
* Add inertia and added mass loads to hydrodynamic and/or buoyant per-unit-length loads

1. Convert the instantaneous fluid acceleration at node to local blade coordinates

are the fluid acceleration terms in local blades coordinates at node

is the orientation matrix of node from AeroDyn [u%BladeMotion(k)%Orientation(:,:,i)]

is the fluid acceleration in global coordinates at the instantaneous position of node , taken as a summation of the current and wave accelerations

1. Calculate per-unit-length inertia forces at node

are the per-unit-length fluid inertia forces in local blades coordinates at node

1. Convert inertia forces in local blade coordinate system to global coordinate system
2. Convert the instantaneous body acceleration at node to local blade coordinates

are the body translational acceleration terms in local blades coordinates at node

is the body translational acceleration in global coordinates at node

is the term of the body rotational acceleration in local blades coordinates at node

is the body rotational acceleration in global coordinates at node

1. Calculate per-unit-length added mass forces at node

are the per-unit-length added mass forces in local blades coordinates at node

1. Calculate per-unit-length added mass pitching moment at node

is the per-unit-length added mass pitching moment in local blades coordinates at node

1. Convert added mass forces and moments in local blade coordinate system to global coordinate system
2. Repeat steps 1-7 for every node
3. Add inertia and added mass loads to hydrodynamic and/or buoyant per-unit-length loads

**Tower**

* Calculate per-unit-length inertia and added mass forces in local tower coordinates at each node according to the appropriate terms from Morison’s equation
* Convert forces to the global coordinate system
* Add inertia and added mass loads to hydrodynamic and/or buoyant per-unit-length loads

1. Convert the instantaneous fluid acceleration at node to local tower coordinates

are the fluid acceleration terms in local tower coordinates at node

is the orientation matrix of node from AeroDyn [u%TowerMotion(k)%Orientation(:,:,i)]

is the fluid acceleration in global coordinates at the instantaneous position of node , taken as a summation of the current and wave accelerations

1. Calculate per-unit-length inertia forces at node

are the per-unit-length fluid inertia forces in local tower coordinates at node

is the fluid density

1. Convert inertia forces in local tower coordinate system to global coordinate system
2. Convert the instantaneous body acceleration at node to local tower coordinates

are the body translational acceleration terms in local tower coordinates at node

is the body translational acceleration in global coordinates at node

1. Calculate per-unit-length added mass forces at node

are the per-unit-length added mass forces in local tower coordinates at node

1. Convert added mass forces in local tower coordinate system to global coordinate system
2. Repeat steps 1-6 for every node
3. Add inertia and added mass loads to hydrodynamic and/or buoyant per-unit-length loads

**UPDATE STATES/END**

Update states and end routines should be unaffected by these changes.