CPEX Particle Data

Version 1.0 (rev 0)

Thomas Hauser

Addition of Spherical Particles to CGNS

Fluid-particulate multiphase flows

Simulating fluid-particle flows has a wide range of applications. One approach for detailed simulations of particle flows is the Discrete Element Model (DEM) where the fluid phase is treated as a continuum and Newton's law of motion is applied to every particle. The DEM approach was introduced by Cundall, P.A. and Strack, O.D. (1979) A discrete numerical model for granular assemblies, Geotechnique, Vol. 21, pp.4765. Fluid-to-particle forces, multi-particle collisions, particlewall collisions and particle body forces are taking into account for the particle movement. The solid and fluid phases are tightly coupled through interphase exchange of momentum and energy in their respective governing equations. A volume fraction of fluid is used in the continuum fluid equations to account for the presence of particulate phase.

Particleparticle and particlewall collision forces are calculated using either a hard sphere model or a soft sphere model. The hard sphere model assumes binary instantaneous collisions between particles at a single point of contact between collision free periods of flight [4]. The soft sphere model on the other hand takes into account a finite collision time with inelastic deformation of the colliding particles with the inclusion of frictional sliding forces.

Need for the Particle Extension

Currently, the CGNS standard does not provide a way to store particle information as part of a solution. This extension proposal tries to address this issue.

Particle Solution `ParticleSolution_t`

This section defines structure types for describing the particle data pertaining to a zone. Entities of each of the structure types defined in this section are contained in the `Zone_t` structure.

Particles and all relevant data within a given zone are described by the `ParticleSolution_t` structure. This structure contains a list for the data arrays of the individual particle variables.

```
ParticleSolution_t<ParticleNumber> :=
  {
  List( Descriptor_t Descriptor1 ... DescriptorN ) ;                    (o)

  List( DataArray_t<DataType, 1, ParticleNumber>
        DataArray1 ... DataArrayN ) ;                                   (o)

  DataClass_t DataClass ;                                               (o)

  DimensionalUnits_t DimensionalUnits ;                                 (o)

  List( UserDefinedData_t UserDefinedData1 ... UserDefinedDataN ) ;     (o)
  } ;
```

Notes

1. Default names for the `Descriptor_t`, `DataArray_t`, and `UserDefinedData_t` lists are as shown; users may choose other legitimate names. Legitimate names must be unique within a given instance of `ParticleSolution_t` and shall not include the names `DataClass`, or `DimensionalUnits`.
2. There are no required fields for `ParticleSolution_t`.
3. The structure parameter `DataType` must be consistent with the data stored in the `DataArray_t` structure entities; `DataType` is `real` for all particle-solution identifiers defined.

`ParticleSolution_t` requires one structure parameters; `ParticleNumber` identifies number of particles.

The particle solution data is stored in the list of `DataArray_t` entities; each `DataArray_t` structure entity may contain a single component of the particle data.

Particle Collision Model: `ParticleCollisionModel_t`

`ParticleCollisionModel_t` describes the collision model used in particle/particle or particle/wall interactions. The enumerated values for `ParticleCollisionModel_t` are a subset of the `ModelType_t` enumeration.

```
ParticleCollisionModelType_t := Enumeration(
```

```
  ModelTypeNull,
  ModelTypeUserDefined,
  Linear,
  NonLinear,
  HardSphere,
  SoftSphere,
  LinearSpringDashpot,
  HertzMindlin,
  HertzKuwabaraKono) ;

ParticleCollisionModel_t :=
  {
  List( Descriptor_t Descriptor1 ... DescriptorN ) ;                    (o)

  ParticleCollisionModelType_t ParticleCollisionModelType ;             (r)

  List( DataArray_t<DataType, 1, 1> DataArray1 ... DataArrayN ) ;       (o)

  DataClass_t DataClass ;                                               (o)

  DimensionalUnits_t DimensionalUnits ;                                 (o)

  List( UserDefinedData_t UserDefinedData1 ... UserDefinedDataN ) ;     (o)
  } ;
```

Notes

1. Default names for the `Descriptor_t`, `DataArray_t`, and `UserDefinedData_t` lists are as shown; users may choose other legitimate names. Legitimate names must be unique within a given instance of `ParticleCollisionModel_t` and shall not include the names `DataClass` or `DimensionalUnits`.
2. `ParticleCollisionModelType` is the only required element.

The following additional variable names are proposed:

CoefNormalRestitution

CoefTangentialRestitution

CoefDynamicFriction

NormalSpringStiffness, k_n $(M/T^2$)

TangentialSpringStiffness, k_t $(M/T^2))$

CoeffSlidingFriction

NormalDampingFactor

TangentialDampingFactor

YoungModulus ($ML^{-1}T^{-2}$)

PoissonRatio

HamakerConstant (ML^2T^{-2})