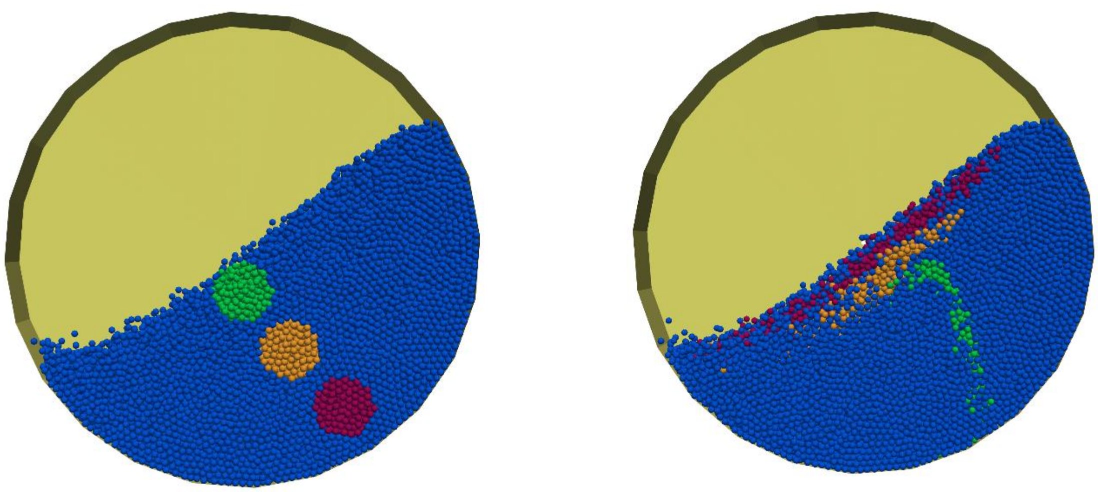
cemfDEM® tutorial

Four: Particle Mixing



Flow pattern and particle mixing in a rotating drum at rolling/cascading regime using DEM

Compatible with gfortran-4.9, intel Fortran 2013, intel Fortran 2015, gfortran-7.5

Author

Hamidreza Norouzi

Amirkabir University of Technology

Center of Engineering and Multiscale Modeling of Fluid Flow

Extra consideration:

- This document is developed to teach how to use cemfDEM® software. The document has gone under several reviews to reduce any possible errors, though it may still have some. We will be glad to receive your comments on the content and error reports through this address: h.norouzi@aut.ac.ir

Document history

Revision	Description	Date
Rev1	Final version	May – 03 – 2021
Rev0	The first version was drafted.	May - 01 - 2021

Table of Contents

1. Problem definition

We investigate mixing mechanism in a rotating drum operating in rolling regime. To this end, a drum with the diameter of 0.2 m and depth of 0.03 m is half-filled with 19000 3-mm spherical particles. Rotation speed of the drum is 10 RPM. This rotation speed creates the cascading regime in the drum. For rolling regime you may decrease the rotation speed. Simulation has three steps: settling of particles in the drum, rotation to reach the steady condition and marking three groups of particles as tracer and tracking them when rotation of drum continues.

> **Note**
>
> **Prerequisites:** If you are not familiar with the code setup, you are recommended to read the previous tutorials in this series. Many of details are covered before.

2. Simulation setup

Download the simulation setup files from the github repository or copy them from "./tutorials/ mixingRotatingDrumRolling" folder into the main root of the source code. Here you first learn how setup the geometry and simulation. After that you will learn how to run it.

2.1. Geometry

The geometry of the rotating drum consists of three entities: one cylindrical shell and two flat planes at two ends. At line 33, a cylindrical shell with radius of 0.1 m and an axis aligned along the z-axis (length is 0.03 m) is created and at line 34, it is added to `Geom` object. The two ends are created with a flat cone and then added to the `Geom` object. The flat cone is a cone whose length is almost zero and the radius of its head is near zero.

Program Listing 1: Code lines of file ProgramDefinedGeometry.f90 for creating drum

```
20      subroutine ProgramDefinedGeometry( Geom )
21      use g_Geometry
22      use g_Line
23      implicit none
```

```
    class(Geometry),intent(in):: Geom

    !// locals
    type(real3) p1, p2, p3, p4
    type(PlaneWall) plane
    type(CylinderWall) cyl,cyl1,cyl2
    logical res

    ! cylinder shell, the radius is 0.1 m
    res =  cyl%CreateCylinder( 0.1_RK, 0.1_RK, p_line( real3(0.0, 0.0, 0.0), real3(0.0,0.0, 0.03) ),24, 1, 1 )
    call Geom%add_Cylinder( cyl )

    ! rear wall
    res =  cyl1%CreateCylinder( 0.001_RK, 0.1_RK, p_line( real3(0.0, 0.0, -0.00001), real3(0.0,0.0, 0.0) ),24, 1, 1 )
    call Geom%add_Cylinder( cyl1 )

    ! front wall
    res =  cyl2%CreateCylinder( 0.1_RK, 0.001_RK, p_line( real3(0.0, 0.0, 0.03), real3(0.0,0.0, 0.03001) ),24, 1, 1 )
    call Geom%add_Cylinder( cyl2 )

end subroutine
```

2.2. Particles and properties

Some code lines of the file main.f90 are listed in Program Listing 2. Note that all the line codes are not presented here. As we have learned in previous tutorials, commands in lines 65 to 87 create 19000 particles, position them in a cuboid (ready for settling), and define the properties for walls and particles. At line 95, DEM system is initialized and at line 98, program iterates for 0.2 seconds (10000 iterations) to let particles settle under their gravity. After the formation of the initial packed bed, two steps should be done: assigning velocity to wall to rotate around drum axis at 10 RPM and marking some particles so that the marked particles can be traced during drum rotations. Line 101 assigns (0,0,0) to translation velocity and 10 RPM to rotation velocity around z-axis for walls with user_id = 1 (this means all three walls created above). At line 106, program iterates for 0.8 so that the bed is transformed into new steady condition. After reaching steady condition (formation of stable bed), some particles are marked at different positions to track their motion during drum rotation. Line 108 invokes the user-defined function in file User_Mark.f90 for all the particles in the simulation (this will be explained in the next section in detail).

Program Listing 2: Code lines of file main.f90 for creating particles and running simulation

```
    PSD  = PS_Distribution( 19000 , PSD_Uniform, 1 , 0.003_RK, 0.003_RK)

    ! associates particle size distribution with property
    allocate(PSDP)
    PSDP = PSD_Property( PSD , prop )
    deallocate(PSD)

    ! positions particles inside the drum
    allocate(Pos)
    Pos = PSDP_Position( PSDP )
    call Pos%PositionOrdered( real3(-0.071 , -0.071, 0.0), real3(0.071, 0.071, 0.03)
)

!// main components of DEM system
    !//// geometry
    allocate(geom)
    call ProgramDefinedGeometry(geom)

    !//// Property
    allocate( Property )
    call Property%ParticleProperty( PSDP )    ! particles
    call Property%WallProperty(1 , (/prop/) ) ! walls
    call Property%PP_BinaryProp(DEM_opt, 0.5_RK, 0.1_RK, 0.8_RK, 0.8_RK) ! binary pp
    call Property%PW_BinaryProp(DEM_opt, 0.7_RK, 0.1_RK, 0.8_RK, 0.8_RK) ! binary pw

    !//// DEM system with particle insertion
    ! simulation domain
    minDomain = real3(-0.11, -0.11,  -0.01)
    maxDomain = real3( 0.11,  0.11 ,  0.04)

    ! initializes DEM system, dt = 0.00002 sec.
    call DEM%Initialize( 0.00002_RK, Pos ,geom, Property, minDomain, maxDomain,
DEM_opt )

!//// iteration loop for 0.2 seconds. This lets particles settle under the gravity
    call DEM%iterate(10000)

    !// sets the rotating velocity of drum, it is 10 RPM
    call geom%setWallVelocity(1, real3(0,0,0), &
                              RPMtoRAD_S(10.0_RK), &
                              p_line( real3(0,0,0) , real3(0,0,0.2) ) )

 !//// iteration for 0.8 more seconds to reach steady condition
    call DEM%iterate(39999)
    !// marks three groups of particles as tracers
    call DEM%User_prtclMark()
    call DEM%iterate(1)

    !/// iteration for 20 more seconds
    call DEM%iterate(1000000)
```

2.3. Marking particle for particle tracking

As we learned earlier, function User_Mark is invoked for all particles in the simulation when statement `call DEM%User_prtclMark()` is executed at line 108 in Program Listing 2. The value of mark is stored in a vector named user_mark in DEM system. Each element of this vector corresponds to one particle in the simulation. So, in this way we can mark particles and tracked these marked particles during simulation.

In Program Listing 3, we see some code lines of User_Mark.90. We define three circles with diameter 0.025 m and centers defined in lines 40-42. Particles whose centers reside in these circles receive a non-zero value as mark and the reset of particles in the simulation receive the value of zero. Later in Figure 1, you will see three circles that are formed in the simulation.

Program Listing 3: Some code lines of file User_Mark.f90

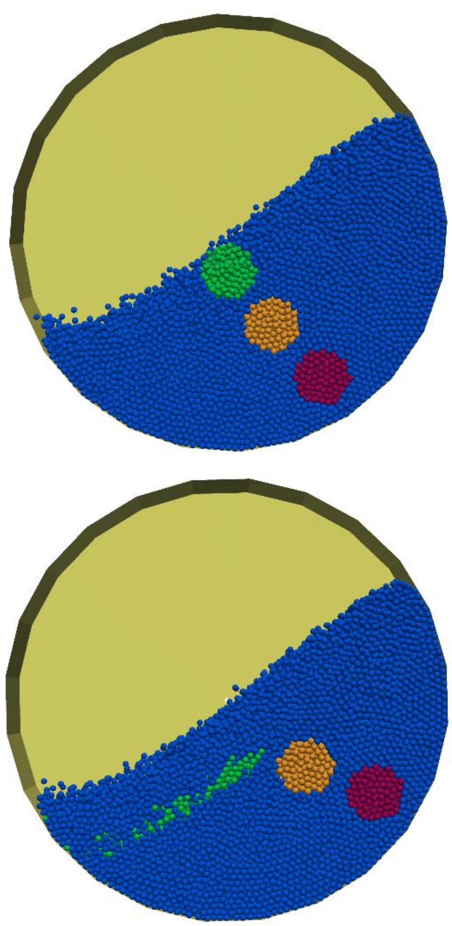
```
20  function User_Mark( id, flag, ptype, dpos, oldMark ) result (mark)

        . . .

35      ! position and diameter of regions which are marked as tracer
36      pos = dpos
37      d1 = 0.025
38      d2 = 0.025
39      d3 = 0.025
40      point1 = real3(0.0, -0.018, 0.0)
41      point2 = real3(0.02, -0.042, 0.0)
42      point3 = real3(0.043, -0.066, 0.0)
43
44      ! finds particles in each region
45      if( sqrt((pos%x- point1%x)**2 + (pos%y- point1%y)**2 ) .le. d1/2) then
46          mark = 1
47      elseif( sqrt((pos%x- point2%x)**2 + (pos%y- point2%y)**2 ) .le. d2/2) then
48          mark = 2
49      elseif( sqrt((pos%x- point3%x)**2 + (pos%y- point3%y)**2 ) .le. d3/2) then
50          mark = 3
51      else
52          mark = 0
53      end if
54
55  end function
```

3. Running simulation (on Ubuntu)

Before building and running the simulation, make sure that all the required files and folders are in the main folder of the program (main.f90, ProgramDefinedGeometry.f90 and User_Mark.f90). Then enter the following commands in the terminal:

```
> make
> ./cemfDEM
```

These commands build the executable file (if no error occurs) and run the simulation. Results of this simulation are visualized using ParaView®. The snapshots in Figure 1 show how the colored particles are moving when the drum is rotating. Flow regime is cascading at this rotation speed.



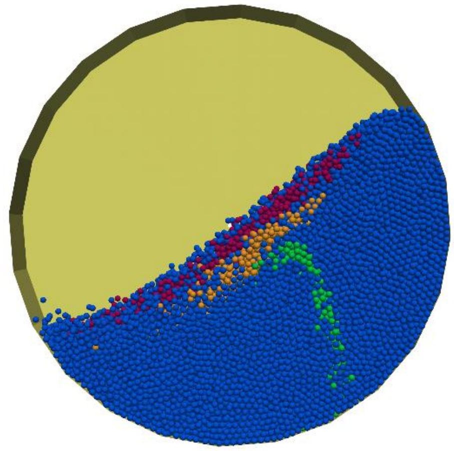


Figure 1: Snapshots of particle motion in a rotating drum.