Documentation for Lipid and Membrane classes

Zoinks! If you are reading this, you might be a tall skinny guy who likes to wear green shirts and have a thing for solving mysteries. I hope the documentation helps and of course Ill be available for any help/discussion. I have already contributed the cosine squared interaction in the official espressomd package, I think they will accept it asap. In case tthey take time you can go to my github page and clone my repository which has the cosine2 interaction.

```
git clone https://github.com/NishchaySuri/espresso
```

1 Using Class Lipid

A Lipid in Cook model is characterised by 3 beads: a head bead and two tail beads.
The FENE and harmonic bond interactions between the lipid beads are automatically set up when we call the add method in the Lipid class.

1.1 Lipid instance

To define a Lipid which has it's mid bead at position (x, y, z), one can simply call the Lipid class from Lipid.py and add it to the system environment in espressomd for integration:

```
from Lipid import Lipid
lipid1 = Lipid(system,midPos = [x,y,z])
lipid1.add()
```

1.2 Tilted Lipid

A lipid can be placed in any orientation in usual spherical polar coordinates by giving a θ and a ϕ angle.

```
from Lipid import Lipid
import numpy as np
lipid2 = Lipid(system,midPos = [x,y,z], theta=np.pi/2, phi=np.pi/4)
lipid2.add()
```

1.3 Using the properties of lipid

Once defined we now know everything about the indivisual beads that form a particular lipid. Each lipid consists of a "Head", "Mid' and a "Tail" bead

```
#To know the positions of all the beads
lipid2.pos
#To know the particleID for all beads
lipid2.partId
#To know the type of all beads
lipid2.type
```

1.4 1-D chain of lipids

We can easily define a 1-D chain of 100 lipids

```
chain = []
position = np.array([0,0,0])
dx = np.array([0.95,0,0]) #vector between 2 lipids

for i in range(100):
        lipid = Lipid(system, position)
        lipid.add()
        chain.append(lipid)
        position += dx
```

2 Using class Membrane

A Membrane consisting of Lipids can be defined similarly by arranging lipids in a specific orientation. We can define a 'monolayer', 'bilayer' or 'mixedbilayer'. We can also put 'random' to randomly put lipids in the box.

2.1 Define a bilayer

```
numLipids = 320 # Number of lipids in the membrane
from Membrane import Membrane
membrane = Membrane(system, numLipids)
membrane.setOrientation('bilayer')

# Setting up non bonded Interactions between the beads
system.non_bonded_inter[0, 0].lennard_jones.set_params(
        epsilon=lj_eps, sigma=0.95 * lj_sig,
        cutoff=lj_cut, shift=1. / 4)

system.non_bonded_inter[1, 1].lennard_jones.set_params(
        epsilon=lj_eps, sigma=lj_sig,
        cutoff=lj_cut, shift=1. / 4)

system.non_bonded_inter[0, 1].lennard_jones.set_params(
        epsilon=lj_eps, sigma=0.95 * lj_sig,
        cutoff=lj_cut_mixed, shift=1. / 4)

# Attractive Tail-Tail
system.non_bonded_inter[1, 1].lennard_jones_cos2.set_params(
        epsilon=lj_eps, sigma=lj_sig,
        width=1.6 * lj_sig, offset=0.)
```

2.2 Define a Mixed Bilayer

We can define a mixed bilayer by defining it's constituting lipid types. The two lipids are of different sizes and we want them to have a different particle type (lipidType) as we do not want the mid beads of the 2 lipid types to have a cosine interaction.

```
lipid1    = Lipid(system, sigma = 0.95,
                          lipidType = {"Head": 0, "Mid": 1, "Tail": 1})
lipid2    = Lipid(system, sigma = 1.05,
                          lipidType = {"Head": 0, "Mid": 2, "Tail": 1})
```

```
membrane = Membrane(system, numLipids)
membrane.setOrientation('mixedbilayer',lipid1=lipid1,lipid2=lipid2)

# Non bonded Interactions between the beads (Have to turn off the Mid-Mid cosine interaction
system.non_bonded_inter[0, 0].lennard_jones.set_params(
                epsilon=lj_eps, sigma=0.95 * lj_sig,
                cutoff=lj_cut, shift=1. / 4)

system.non_bonded_inter[0, 1].lennard_jones.set_params(
                epsilon=lj_eps, sigma=0.95 * lj_sig,
                cutoff=lj_cut_mixed, shift=1. / 4)

system.non_bonded_inter[0, 2].lennard_jones.set_params(
                epsilon=lj_eps, sigma=0.95 * lj_sig,
                cutoff=lj_cut_mixed, shift=1. / 4)

system.non_bonded_inter[1, 1].lennard_jones.set_params(
                epsilon=lj_eps, sigma=lj_sig,
                cutoff=lj_cut, shift=1. / 4)

system.non_bonded_inter[2, 1].lennard_jones.set_params(
                epsilon=lj_eps, sigma=lj_sig,
                cutoff=lj_cut, shift=1. / 4)

system.non_bonded_inter[2, 2].lennard_jones.set_params(
                epsilon=lj_eps, sigma=lj_sig,
                cutoff=lj_cut, shift=1. / 4)

# Attractive Tail-Tail
system.non_bonded_inter[1, 1].lennard_jones_cos2.set_params(
                epsilon=lj_eps, sigma=lj_sig,
                width=1.6 * lj_sig, offset=0.)

system.non_bonded_inter[2, 2].lennard_jones_cos2.set_params(
                epsilon=lj_eps, sigma=lj_sig,
                width=1.6 * lj_sig, offset=0.)
```

3 Accessing any Lipid

We can access any lipid in a membrane with membrane.lipid[number] which will return a lipid instance. That will hold everything about the beads and we can use all the properties of the class Lipid to use them. The most important property is the partId which is how espressomd recognizes a particular particle and has all the information on it.

```
#The particle Id's for Head, Mid and Tail for lipid 105 in the membrane
headId = membrane.lipid[105].partId['Head']
midId =membrane.lipid[105].partId['Mid']
tailId = membrane.lipid[105].partId['Tail']

#An example to return the position and velocity
system.part[headId].pos
system.part[headId].v
```

4 General things

It is upto the group on how all of them want to proceed extending the class Lipid and Membrane. If in the project you find it easier to write your own membrane methods or just use Lipid class to place lipids forming membranes yourself that will be also a good way to go. I have written 2 tutorials on how to simulate a bilayer and a mixed bilayer(with visualization) and are in the git repsitory BioPhys. To clone it just do :

```
git clone https://github.com/NishchaySuri/BioPhys
```