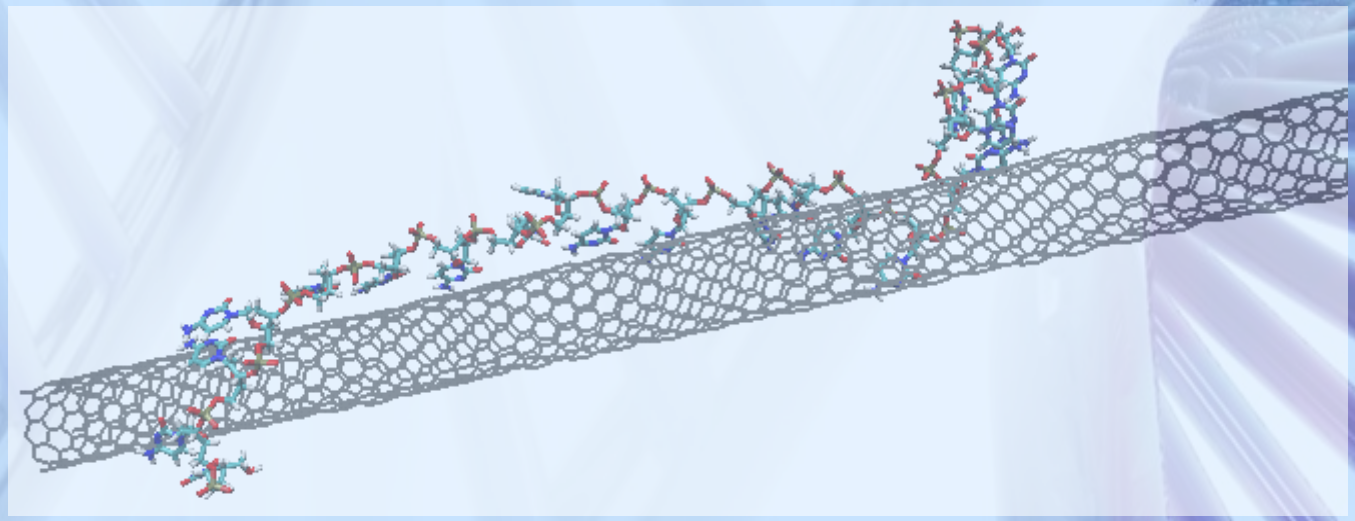


A Model building, Simulation and Data Analysis Script for the Study of DNA-CNT Hybrids

Myrna I. Merced Serrano
Department of Mathematics
University of Puerto Rico at Humacao

DNA-CNT Hybrids

Single Stranded DNA

- DNA molecule consisting of only one chain of alternating sugars and phosphates.
- For testing purpose we use a ss-DNA composed of a repeating sequence of cytosines (Poly-C ss-DNA).

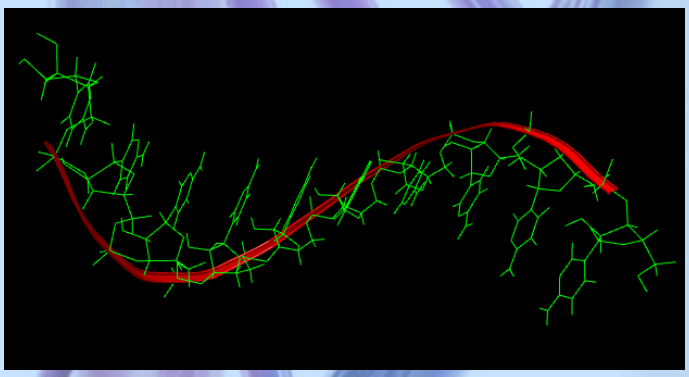


Figure 1. Poly-C ss-DNA

DNA-CNT Hybrids

Carbon Nanotubes

- cylindrical sheets of carbon
- have diameters of ~1nm and lenghts up to a few centimeters
- have structural and electrical properties
- sensors

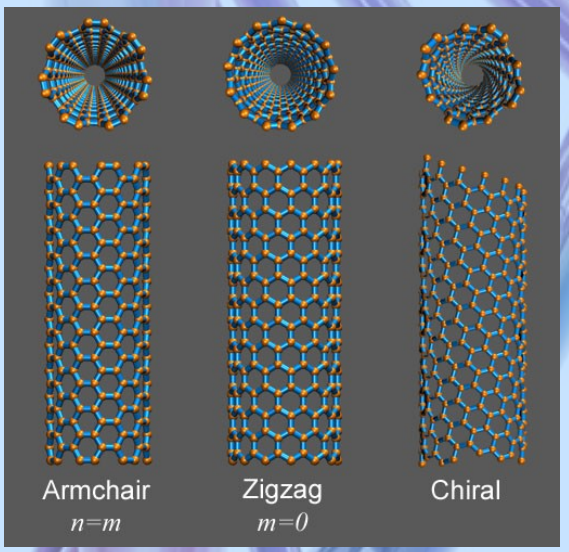
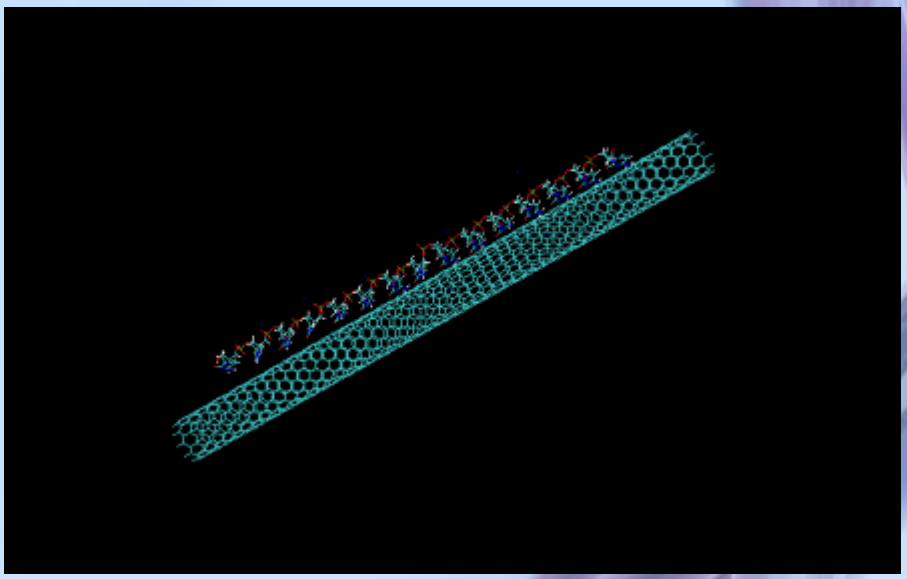


Figure 2. Different forms of CNTs.

Problem

To develop computational tools for molecular dynamics simulations between a ss-DNA and a CNT.

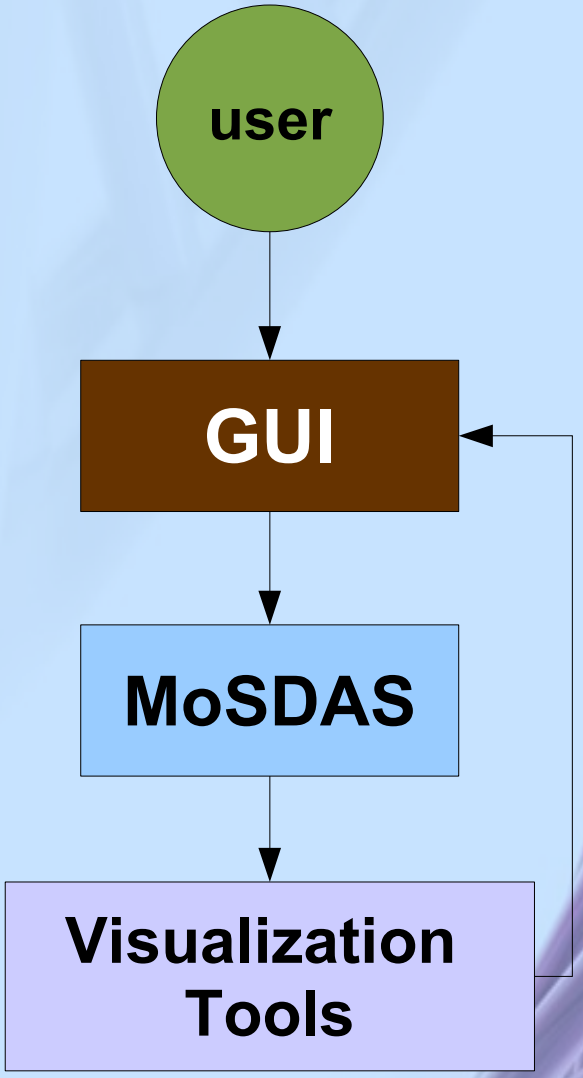


Software and Methods

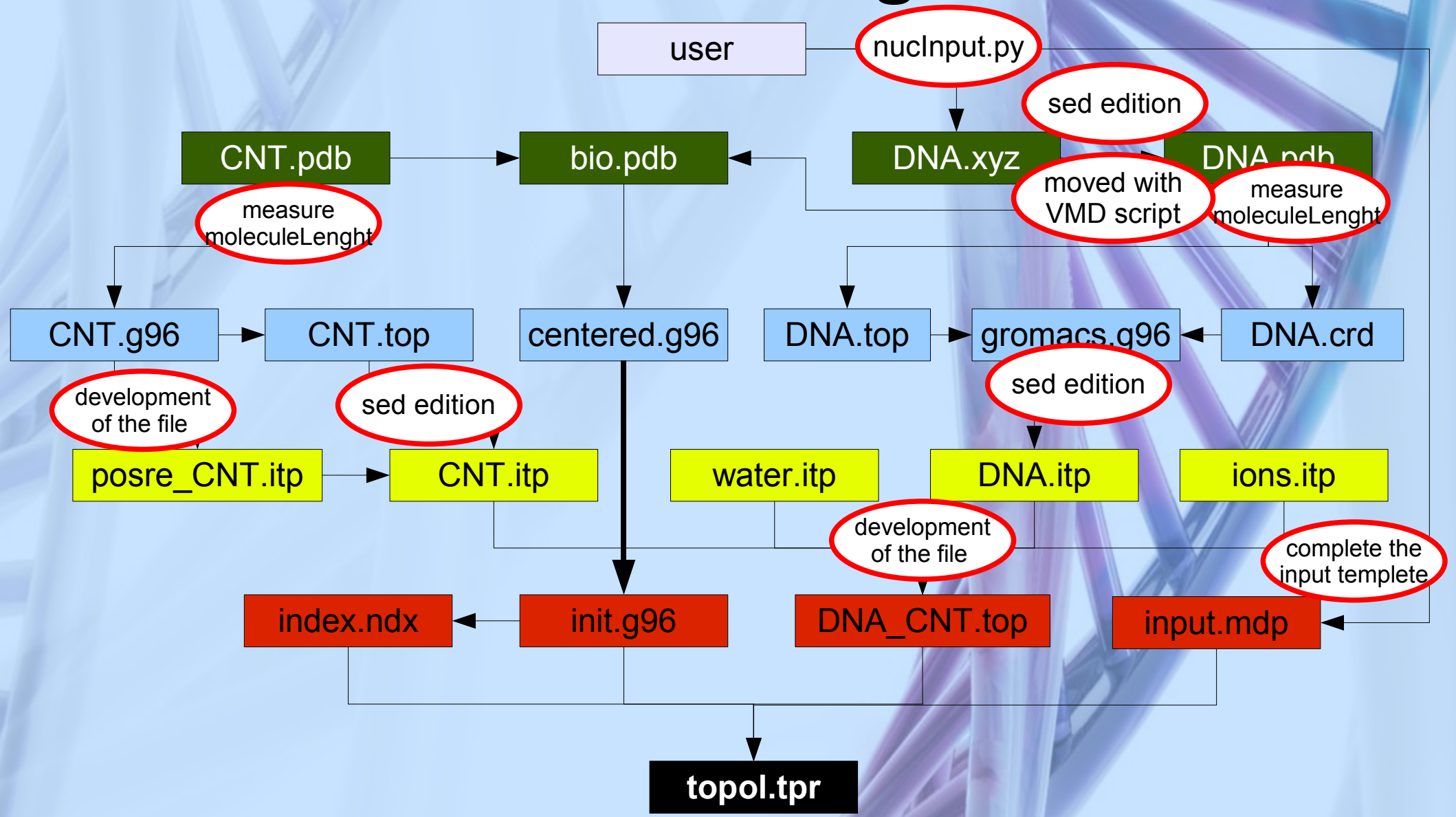
- ***Software to be integrated***
  - GROMACS MD package
  - Tinker molecular modeling package
  - AMBER7 molecular dynamics package
  - Visual Molecular Dynamics (VMD)

- ***Methods***
  - Development of the *Model building, Simulation and Data Analysis Script* (MoSDAS).
  - Development of a Graphic User Interface and integrate MoSDAS to it.

Methods Diagram



MoSDAS Diagram



MoSDAS Diagram

topol.tpr

runGromacs.sh

simulation

DNA_CNT.g96

DNA_CNT.trr

analysis

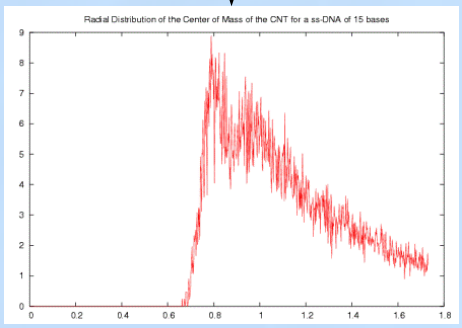
development of a program

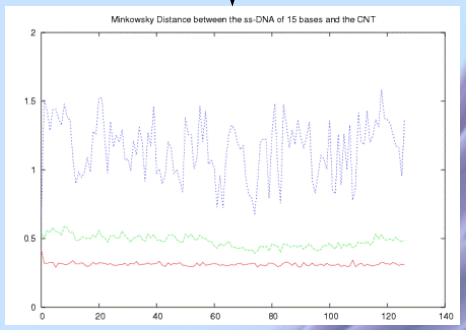
development of a program

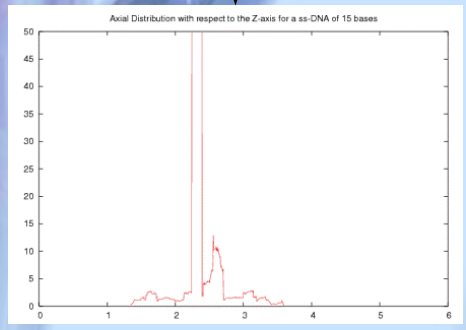
Radial Distribution

Molecule Separation

Axial Distribution



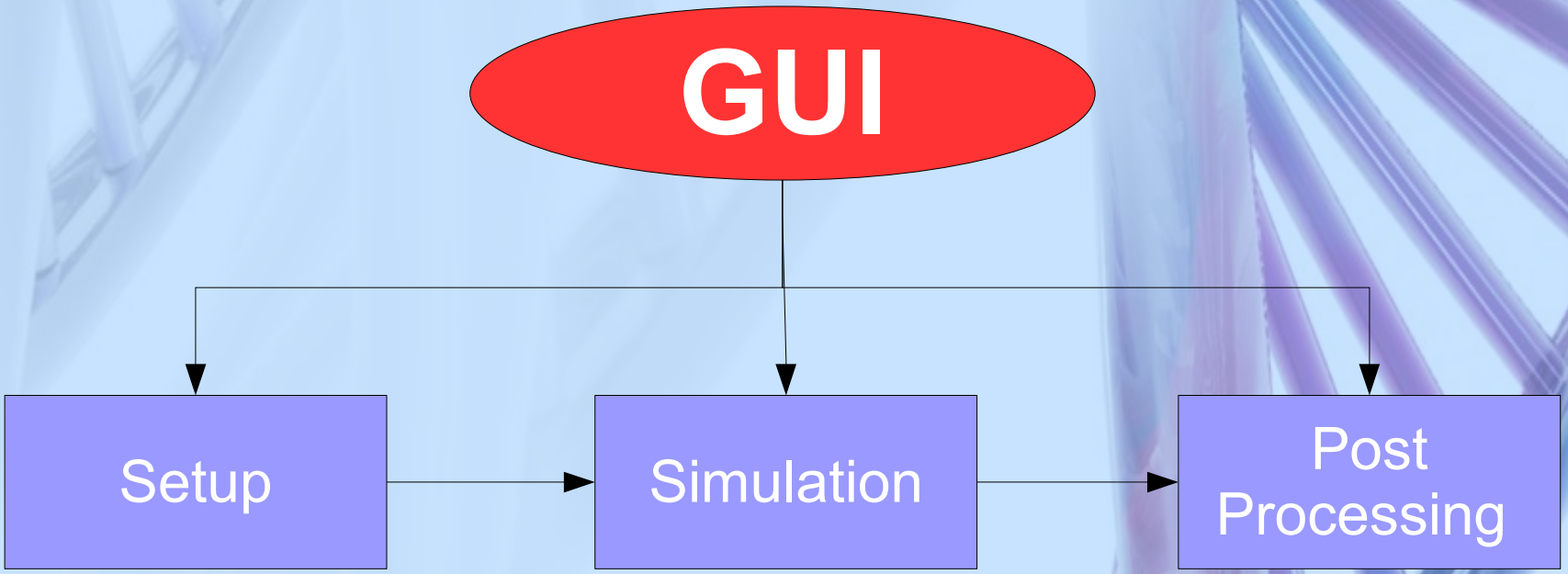




MoSDAS Code

```
#!/bin/bash

PROGS_DIR=/home/mymese/inv/DNA_Shell
PARAM_DIR=/home/mymese/inv/tinker/params
TINKER_BIN_DIR=/home/mymese/inv/tinker/bin
CNT_DIR=/home/mymese/inv/nanotubegen
GROMACS_DIR=/usr/local/gromacs/bin

if test $2 = min
then

#generate the ssDNA
$PROGS_DIR/nucInput.py $1 | $TINKER_BIN_DIR/nucleic
rm DNA_$1L.pdb
echo "$PARAM_DIR/amber94.prm" | $TINKER_BIN_DIR/xyzpdb DNA_$1L.xyz

#prepare pdbfile for xleap.  Replace CYTs DCs for codes that xleap can understand and
delete H atoms
/bin/sed -n -e s/CYT/DC/ -e "w DNA_$1L.tmp"  DNA_$1L.pdb
/bin/sed -e "s/DC     1 /DC5    1 /" -e "s/DC    $1 /DC3   $1 /" -e "s/DC     $1 /DC3    $1 /"
-e /H/d -e "/O2\*/d" DNA_$1L.tmp > DNA_$1L.pdb
rm  DNA_$1L.tmp

#move the cordinates of the DNA
ADD_TO_TUBE=30
ADD_TO_TUBE2=`echo "scale=3; $ADD_TO_TUBE / 2" | /usr/bin/bc`

echo -e "set D [atomselect 0 \"all\"]\n \$D move {{0.469471563 -0.879826666 0 0}
{0.879826666 0.475294686 0 0} {0 0 1 0} {0 0 0 1}}\n  \$D move {{1 0 0 0} {0
0.669130606 -0.743144825 0} {0 0.743144825 0.669130606 0} {0 0 0 1}}\n \$D moveby
{-5.7 5.7 $ADD_TO_TUBE2}\n \$D writepdb DNA_$1L.pdb\n quit\n" | /usr/local/bin/vmd
-dispdev text DNA_$1L.pdb

#add ions and generate the topology file with tleap
IONS=`echo "$1-1" | /usr/bin/bc`

echo -e "dna = loadpdb "DNA_$1L.pdb"\n addions dna Na+ $IONS\n savePDB dna
DNA_$1L.pdb \n saveamberparm dna DNA_$1L.top DNA_$1L.crd\n quit\n" >
DNA_$1L.tleap

tleap -f DNA_$1L.tleap

#measure the DNA lenght
DNA_LENGHT=`$PROGS_DIR/moleculeLength DNA_$1L.pdb`
echo "Largo del DNA: $DNA_LENGHT"

#convert the coodinates and top file to GROMACS format
Conv_7.x DNA_$1L.top DNA_$1L.crd

#edit the DNA topology file
SYS_LINE_DNA=`grep -n "\[ system"  gromacs.top | cut -d ':' -f1`

/bin/sed -e "s/Protein/DNA_$1L/" -e "/#include/d" -e "s/^\*/" -e  "$SYS_LINE_DNA,\$d"
gromacs.top >  DNA_$1L.itp

#calculate the tube lenght
CNT_LENGHT_AMS=`echo $ADD_TO_TUBE + $DNA_LENGHT | /usr/bin/bc`
echo "Largo del tubo en amstrongs: $CNT_LENGHT_AMS"

#generate the tube
echo "$CNT_LENGHT_AMS 11 0"  | $CNT_DIR/nanotubegen

#measure the CNT lenght
#CNT_LENGHT=`$PROGS_DIR/moleculeLength nanotube.pdb`
CNT_LENGHT=`echo -e "set CNT [atomselect 0 \\"all\\"]\\n measure minmax \\$CNT\\n
quit" | /usr/local/bin/vmd -dispdev text nanotube.pdb | tail -3 |head -1 | cut -d' ' -f6 | cut
-c1-6`

CNT_LENGHT=`echo "scale=3; $CNT_LENGHT / 10.00" | /usr/bin/bc`
echo "Largo del tubo en nano antes de la suma: $CNT_LENGHT"

#set the correct lenght of the tube
ADD_TO_CNT=0.1418*0.5
CNT_LENGHT=`echo $CNT_LENGHT + $ADD_TO_CNT | /usr/bin/bc`
echo "Largo del tubo en nanometros despues de la suma: $CNT_LENGHT"

#rename the CNT pdb file
mv nanotube.pdb CNT_$CNT_LENGHT.pdb

#center the CNT in a box
$GROMACS_DIR/editconf -f CNT_$CNT_LENGHT.pdb -o CNT_$CNT_LENGHT.g96 -c
-box 10 10 $CNT_LENGHT

#generate the CNT topology file
echo "7" | $GROMACS_DIR/x2top -f CNT_$CNT_LENGHT.g96 -o
CNT_$CNT_LENGHT.top -pbc -nopairs -name CNT_$CNT_LENGHT -nexcl 5

#edit the CNT top file
BOND_LINE=`grep -n "\[ bond"  CNT_$CNT_LENGHT.top | cut -d ':' -f1`
ANGLES_LINE=`grep -n "\[ angles"  CNT_$CNT_LENGHT.top | cut -d ':' -f1`
DIHE_LINE=`grep -n "\[ dihedra"  CNT_$CNT_LENGHT.top | cut -d ':' -f1`
SYS_LINE=`grep -n "\[ system"  CNT_$CNT_LENGHT.top | cut -d ':' -f1`

/bin/sed  -e "$BOND_LINE,$ANGLES_LINE s/1 \$/3 /" -e "$ANGLES_LINE,$DIHE_LINE s/1 \$/2 /" -e "s/      C          0          0   ; qtot 0//" -e "/#include/d" -e "$SYS_LINE,\$d"
CNT_$CNT_LENGHT.top > CNT_$CNT_LENGHT.tmp

echo -e ";#ifndef POSRES_CNT_$CNT_LENGHT \n#include \"posre_CNT_$CNT_LENGHT.itp\"\n;#endif" >> CNT_$CNT_LENGHT.tmp
mv CNT_$CNT_LENGHT.tmp CNT_$CNT_LENGHT.itp

#join tube and ssDNA
echo -e "dna = loadpdb "DNA_$1L.pdb"\n cnt = loadpdb "CNT_$CNT_LENGHT.pdb"\n bio = combine {dna, cnt}\n savePDB bio bio_$1L.pdb\n quit\n" > bio_$1L.tleap

tleap -f bio_$1L.tleap

#hidratate the system
$GROMACS_DIR/editconf -f bio_$1L.pdb -o center_$1L.g96 -c -box 5 5 $CNT_LENGHT
$GROMACS_DIR/genbox -cp center_$1L.g96 -cs /home/mymese/inv/penn/topology/spc216.gro -o init_$1L.g96

#remove the water and ions inside of the tube
echo -e "source /home/mymese/inv/penn/scripts/rem-wat-interior.tcl" | /usr/local/bin/vmd -dispdev text init_$1L.g96

$GROMACS_DIR/editconf -f hollow.pdb -o init_$1L.g96

#make the index file
echo -e "r 1-$1\n name 7 DNA_$1L\n\nq\n" | $GROMACS_DIR/make_ndx -f init_$1L.g96 -o index_$1L.ndx

# count atoms in system
CNT_COUNT=`grep CNT init_$1L.g96 | wc -l`
WAT_COUNT=`grep WAT init_$1L.g96 | wc -l`
WAT_COUNT=`echo $WAT_COUNT / 3 | /usr/bin/bc`
NA_COUNT=`grep "Na+" init_$1L.g96 | wc -l`
DNA_COUNT=`grep DC init_$1L.g96 | wc -l`

#  creates position restraints of CNT
echo "[ position_restraints ]" > posre_CNT_$CNT_LENGHT.itp
awk "BEGIN{atom_num=$CNT_COUNT; fx = 1000; fy = 1000; fz = 1000;for(i=1;i<=atom_num;i++) print i, \"1\", fx, fy ,fz}" >> posre_CNT_$CNT_LENGHT.itp

# create topology file
awk "BEGIN{print \"\
; Topology for CNT - DNA FET (generated by \"automatizacion\" script).\n\
#include \\\"/home/mymese/inv/penn/topology/ffbiosensor.itp\\\"\n\n\
;Topology include files\n\
#include \\\"DNA_$1L.itp\\\"\n\
#include \\\"CNT_$CNT_LENGHT.itp\\\"\n\
#include \\\"/home/mymese/inv/penn/topology/water.itp\\\"\n\
#include \\\"/home/mymese/inv/penn/topology/na_ion.itp\\\"\n\n\
[ system ]\n\
DNA_CNT_$1L\n\n\
[ molecules ]\n\
; name        number\n\
DNA_$1L                 1\n\
CNT_$CNT_LENGHT         1\n\
water                   $WAT_COUNT\n\
;na_ion                 $NA_COUNT\n\
\"}"> DNA_CNT_$1L.top

#complete the templete for the input file
$PROGS_DIR/shellPruebaInput $1 $2 $3

#remove the not needed files
rm gro* \#* *.crd *.int *.seq *.tleap *.xyz *.log hollow.pdb ffgro96.itp

elif test $2 = cont
then

#complete the templete for the input file
$PROGS_DIR/shellPruebaInput $1 $2 $3

fi
```

MoSDAS GUI Diagram

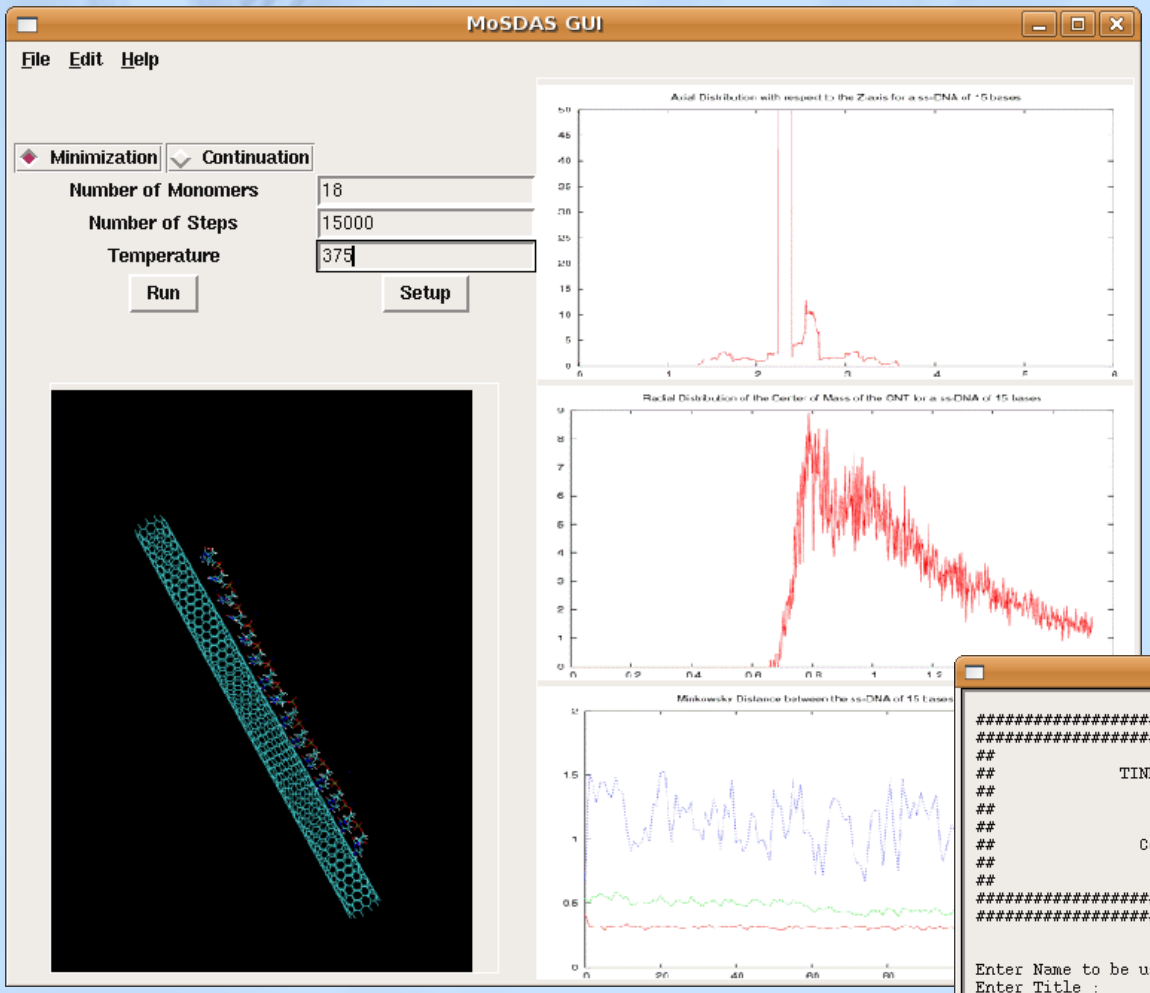


MoSDAS GUI Code

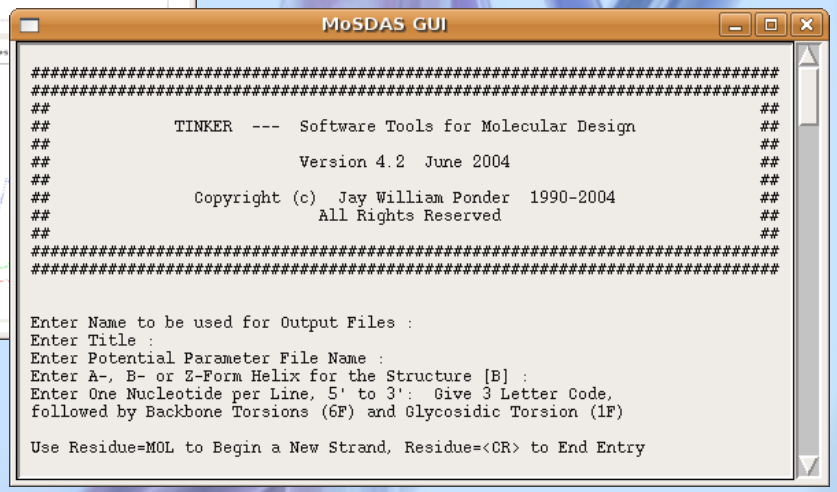
```
# Frame-based menus: for top-levels and components
import sys, time, os, launchmodes, ImageTk
from Tkinter import *                           # get widget classes
from tkMessageBox import *                      # get standard dialogs

if __name__ == '__main__':
    root = Tk()                                  # or TopLevel or Frame
    root.title('Interfaz_prueba')                # set window-mgr info
    gifDir="./"
    igm = PhotoImage(file=gifDir+"snap.gif")
    ADF_igm = PhotoImage(file=gifDir+"ADF_15L.gif")
    RDF_igm = PhotoImage(file=gifDir+"RDF_15.gif")
    PointSet_igm = PhotoImage(file=gifDir+"Point-Set_15.gif")

#=============================================================================
class RunButton(Button):
    def exucuteSimulation(x):
       showerror('exucuteSimulation', 'exucuteSimulation')

    def __init__(self, aFrame):
       Button.__init__(self, aFrame, text='Run', command=self.exucuteSimulation)

#=============================================================================
class SetupButton(Button):
    def runMOSDAS(x):
       showerror('runMOSDAS', 'runMOSDAS')
       #botToActivate.activate()

    def __init__(self, aFrame, otherButton):
       Button.__init__(self, aFrame, text='Setup', command=self.runMOSDAS)
       botToActivate = otherButton

#=============================================================================
class MinContRadiobutton(Radiobutton):

  radiovar = StringVar()

  def muestraSeleccion():
    showerror(self.radiovar.get(), self.radiovar.get())

  def __init__(self, parent):
    pointBotton = Frame(parent)
    pointBotton.grid(row=0, column=0)

    links = ["Minimization", "Continuation"]
    cont=-1
    for (name) in links:
            cont = cont+1
            link = Radiobutton(pointBotton, text=name)
            link.config(relief=GROOVE, variable=self.radiovar, value=name,
command=self.muestraSeleccion)
            #link.pack(side=LEFT, expand=YES, fill=BOTH)
        link.grid(row=4, column=cont )

#=============================================================================
class LoadPhoto:
   def __init__(self, parent, imagen):
      can=Canvas(parent)
      can.create_image(1,1,image=imagen, anchor=NW)
      can.pack(side=RIGHT, fill=Y)

#=============================================================================
class LoadPhoto2:
   def __init__(self, parent, imagen):
      can=Canvas(parent)
      can.create_image(1,1,image=imagen, anchor=NW)
      can.pack(side= RIGHT, fill=Y)

#=============================================================================
class EntryLabel:
  def __init__(self, parent):
    label1= Label(parent, text="Number of Monomers")
    label1.grid(row=1, column=0)
    entry1= Entry(parent)
    entry1.grid(row=1, column=1)

    label2= Label(parent, text="Number of Steps")
    label2.grid(row=2, column=0)
    entry2= Entry(parent)
    entry2.grid(row=2, column=1)

    label3= Label(parent, text="Temperature")
    label3.grid(row=3, column=0)
    entry3= Entry(parent)
    entry3.grid(row=3, column=1)
```

```
#=============================================================================
class ButtonsArea:
  def  __init__(self, parent):
    areaBotones = Frame(parent)
    areaBotones.pack()

    rbuton = RunButton(areaBotones)
    rbuton.grid(row=4, column=0)
    SetupButton(areaBotones, rbuton).grid(row=4, column=1)
    EntryLabel(areaBotones)
    MinContRadiobutton(areaBotones)

#=============================================================================
class MakeMenu:
  def notdone(x):
    showerror('Not implemented', 'Not yet available')

  def __init__(self, parent):
    menubar = Frame(parent)
    menubar.pack(side=TOP, fill=X)

    fbutton = Menubutton(menubar, text='File', underline=0)
    fbutton.pack(side=LEFT)
    file = Menu(fbutton, tearoff=0)

    file.add_separator()
    submenu = Menu(file, tearoff=0)
    submenu.add_command(label='ADN', command=parent.quit, underline=0)
    submenu.add_command(label='PEO', command=self.notdone, underline=0)
    file.add_cascade(label='New', menu=submenu, underline=0)

    file.add_command(label='Open...', command=self.notdone, underline=0)
    file.add_command(label='Save...', command=self.notdone, underline=0)
    file.add_command(label='Export...', command=self.notdone, underline=0)
    file.add_command(label='Quit', command=parent.quit, underline=0)
    fbutton.config(menu=file)

    ebutton = Menubutton(menubar, text='Edit', underline=0)
    ebutton.pack(side=LEFT)
    edit = Menu(ebutton)
    edit.add_command(label='Cut', command=self.notdone, underline=0)
    edit.add_command(label='Paste', command=self.notdone, underline=0)
    ebutton.config(menu=edit)

    hbutton = Menubutton(menubar, text='Help', underline=0)
    hbutton.pack(side=LEFT)
    help = Menu(hbutton)
    help.add_command(label='About...', command=self.notdone, underline=0)
    hbutton.config(menu=help)

#=============================================================================

if __name__ == '__main__':
    MakeMenu(root)                          # associate a menu bar
    ButtonsArea(root)

    label1= Label(root, text="Axial distribution")
    label1.pack()
    LoadPhoto2(root, ADF_igm)

    label2= Label(root, text="Radial distribution")
    label2.pack()
    LoadPhoto2(root, RDF_igm)

    label3= Label(root, text="Point-to-set distances")
    label3.pack()
    LoadPhoto2(root,PointSet_igm)

    label4= Label(root, text="Initial Position of the Atoms")
    label4.pack()
    LoadPhoto(root, igm)


    root.mainloop()
```

Results



Prototype of MoSDAS GUI



Results and Conclusions

Results

- The system's setup time was reduced from a day to 10 seconds.
- MoSDAS improved the analysis of data.

Conclusions

- MoSDAS avoids the risk of errors on the simulation process.
- We can measure conformation aspects of the DNA wraping around the CNT.
- The prototype of MoSDAS GUI is a great help for run MoSDAS.
- MoSDAS GUI can help to have a better data organization.

Acknowledgments

- PREM (NSF-DMR-353730)
- URMAA (NSA-H98230-04-C-0486)
- UPRH RISE
- CSEMS(NSF-0123169)

References

1. Merced M., "*Automatization of a molecular dynamics simulation and the evaluation of metrics for the study of DNA-CNT Hybrids*", Dec. 2006

2. Zheng M., et al., "*Structure-based carbon nanotube sorting by sequence-dependent DNA assembly.*", Science, 2003 Nov 28; 302(5650): 1545-8.

3. Johnson Group: Experimental Nanoscale Physics-Resources, webside: http://www.lrsm.upenn.edu/~nanophys/nanotube.html