Listing of file drawappldothework23a.h

Ulrich Mutze
www.ulrichmutze.de

July 25, 2009

1 drawappldothework23a.h

```
//? drawappldothework23a.h
//? C+- by Ulrich Mutze. Status of work 2009-07-25.
//? Copyright (c) 2009 Ulrich Mutze, www.ulrichmutze.de
//? All rights reserved

// this is the common expression of the LineArt definition
// which will be worked through in 2D and 3D

   Frame fr;
   Word sec("camera control");
   Camera ca(rch,sec,fr);
   ca.clr();
   sec="run control";
   R tWait=0;
   bool lineArtProcessing=false;
   cpmrh(tWait);
   cpmrh(lineArtProcessing);
// Basic reference quantities
   Spc o;
   V<Vec> b=Vec::canBas();
   Vec e1=b[1];
   Vec e2=b[2];
   Vec e3=b(3);

   Vec tv1=e1;
   Vec tv2=e2;
   Spc t1=o;
   Spc t2=t1+tv1;
   Spc t3=t2+tv2;
   Spc t4=t1+tv2;
   V<Spc> vtb("",t1,t2,t3,t4);
   Skeleton tb1(vtb,true);
      // table 1
   tb1.clsPath_();
   Skeleton tb2(tb1);
   Skeleton tb3(tb1);

   V<Word> vw0("","t","i","m","e");
   V<Word> vw1("","s","p","a","c","e");
   V<Word> vw2("","s","t","a","t","e");
   V<Word> vw3("","i","n","i","t","i","a","l","space");
   V<Word> vw4("","f","i","n","a","l","space");
   V<Word> vw5("","i","n","t","e","r","m","e","d","i","a");
   V<Word> vw5a("","t","e","space");

   V<Word> ax1("","t");
   V<Word> ax2("","t","plus","d","t","slash","n2");
   V<Word> ax3("","t","plus","d","t");

// position in initial state
   Wrt xi("x");
   V<Wrt> index("",Wrt("i"));
   xi.sub_(index);
   LineArt lxi(xi);

// position in final state
   Wrt xis("x");
   V<Wrt> indexU("",Wrt("ast"));
   xis.sub_(index);
   xis.sup_(indexU);
   LineArt lxis(xis);

// creating the necessary indexes

   V<Wrt> ind_1i("",Wrt("n1"),Wrt("hs"),Wrt("i"));
   V<Wrt> ind_2i("",Wrt("n2"),Wrt("hs"),Wrt("i"));
   V<Wrt> ind_3i("",Wrt("n3"),Wrt("hs"),Wrt("i"));

   V<Wrt> ind_2l("",Wrt("n2"),Wrt("hs"),Wrt("l"));
   V<Wrt> ind_2k("",Wrt("n2"),Wrt("hs"),Wrt("k"));
   V<Wrt> ind_2j("",Wrt("n2"),Wrt("hs"),Wrt("j"));


// points representing the intermediate state

   Wrt y1i("y");
   y1i.sub_(ind_1i);
   LineArt ly1i(y1i);

   Wrt y2i("y");
   y2i.sub_(ind_2i);
   LineArt ly2i(y2i);

   Wrt y3i("y");
   y3i.sub_(ind_3i);
   LineArt ly3i(y3i);

   Wrt y2l("y");
   y2l.sub_(ind_2l);
   LineArt ly2l(y2l);

   Wrt y2k("y");
   y2k.sub_(ind_2k);
   LineArt ly2k(y2k);

   Wrt y2j("y");
   y2j.sub_(ind_2j);
   LineArt ly2j(y2j);

// vectors representing the velocities of the initial state

   Wrt v1i("v");
   v1i.vec_();
   v1i.sub_(ind_1i);
   LineArt lv1i(v1i);

   Wrt v2i("v");
   v2i.vec_();
   v2i.sub_(ind_2i);
   LineArt lv2i(v2i);

   Wrt v3i("v");
   v3i.vec_();
   v3i.sub_(ind_3i);
   LineArt lv3i(v3i);

// vectors representing the velocities of the final state

   Wrt v1is("v");
   v1is.vec_();
   v1is.sub_(ind_1i);
   v1is.sup_(indexU);
   LineArt lv1is(v1is);

   Wrt v2is("v");
   v2is.vec_();
   v2is.sub_(ind_2i);
   v2is.sup_(indexU);
   LineArt lv2is(v2is);

   Wrt v3is("v");
   v3is.vec_();
   v3is.sub_(ind_3i);
   v3is.sup_(indexU);
   LineArt lv3is(v3is);

   V<Word> tt1=vw3&vw2; // initial state
   V<Word> tt2=vw5&vw5a&vw2; // intermediate state
   V<Word> tt3=vw4&vw2; // final state

   LineArt time(vw0);
   LineArt space(vw1);
   LineArt inistate(tt1);
   LineArt intstate(tt2);
   LineArt finstate(tt3);
   LineArt lax1(ax1);
   LineArt lax2(ax2);
   LineArt lax3(ax3);

   sec="text";
   R sclText=1;
   Z nThick=0;
   cpmrh(sclText);
   cpmrh(nThick);

// rescaling all text elements
   time.scl_(o,sclText);
   space.scl_(o,sclText);
   inistate.scl_(o,sclText);
   intstate.scl_(o,sclText);
   finstate.scl_(o,sclText);
   lax1.scl_(o,sclText);
   lax2.scl_(o,sclText);
   lax3.scl_(o,sclText);
   lxi.scl_(o,sclText);
   lxis.scl_(o,sclText);
   lv1i.scl_(o,sclText);
   lv2i.scl_(o,sclText);
   lv3i.scl_(o,sclText);
   lv1is.scl_(o,sclText);
   lv2is.scl_(o,sclText);
   lv3is.scl_(o,sclText);
   ly1i.scl_(o,sclText);
   ly2i.scl_(o,sclText);
   ly3i.scl_(o,sclText);
   ly2l.scl_(o,sclText);
   ly2k.scl_(o,sclText);
   ly2j.scl_(o,sclText);

   Z i;
   LineArt textExample=inistate;
   R letterHeight=textExample.h();

   sec="system data";
   R nameShift_x=0.05;
   R nameShift_y=0.05;
   cpmrh(nameShift_x);
   cpmrh(nameShift_y);

   LineArt spaceMem=space;
   Vec namShift(nameShift_x,nameShift_y);

// creating the table for the initial state
   LineArt tab1;
   tab1&=tb1;

   R letterShift=0.05;
   R verticalRelTextShift=1.25;
   cpmrh(letterShift);
   cpmrh(verticalRelTextShift);

   Vec sh1(letterShift,1+verticalRelTextShift*letterHeight);
   Vec sh2(letterShift,verticalRelTextShift*letterHeight);
   inistate+=sh1;
   space+=sh2;
   tab1&=inistate;
   tab1&=space;
   R tw=1,th=1;
// main symbol on tab1
   R xi1=0.5;
   R xi2=0.75;
   cpmrh(xi1);
   cpmrh(xi2);
   Spc pxi(xi1,xi2);
   R rD=0.015;
   cpmrh(rD);
   LineArt d11=LineArt::disk(pxi,e3,rD);
   tab1&=d11;
   lxi.to_(pxi);
   lxi+=namShift;
   lxi+=Vec(0,letterHeight);
   tab1&=lxi;
   Group g1(o,AxVec(Angle(90,DEG),e2));
   tab1*=g1;
   pxi*=g1;
   lv1i*=g1;
   lv2i*=g1;
   lv3i*=g1;

   R timeStep=2;

// creating the table for the intermediate state

   LineArt tab2;
   tab2&=tb2;
   intstate+=sh1;
   LineArt space2=spaceMem;
   space2+=sh2;
   tab2&=intstate;
   tab2&=space2;

// main symbols on table 2
// We need 6 points on the plane.
// y1i,y2i,y3i,y2l,y2k,y2j, the points indexed i should
// approximately form an equilateral triangle.

   V<LineArt> yNam("",ly1i,ly2i,ly3i,ly2l,ly2k,ly2j);
```

```
Z ny=yNam.dim();
V<Spc> pos(ny);
R yc1=0.55, yc2=0.6;
cpmrh(yc1);
cpmrh(yc2);
Spc yc(yc1,yc2);
R rc=0.25,phic=20;
cpmrh(rc);
cpmrh(phic);
Vec vc1(0,rc);
vc1*=AxVec(Angle(phic,DEG));
Vec vc2=vc1;
AxVec a120(Angle(120,DEG));
vc2*=a120;
Vec vc3=vc2;
vc3*=a120;
pos[1]=yc+vc1;
pos[2]=yc+vc2;
pos[3]=yc+vc3;

R p41=0.21,p42=0.23,p51=0.83,p52=0.21,p61=0.8,p62=0.82;
cpmrh(p41);
cpmrh(p42);
cpmrh(p51);
cpmrh(p52);
cpmrh(p61);
cpmrh(p62);

pos[4]=Spc(p41,p42);
pos[5]=Spc(p51,p52);
pos[6]=Spc(p61,p62);

for (i=1;i<=ny;++i){
   tab2&=LineArt::disk(pos[i],e3,rD);
   yNam[i].to_(pos[i]);
   yNam[i]+=namShift;
   if (i==5) yNam[i]+=Vec(-0.15*letterHeight,0);
   tab2&=yNam[i];
}
R omegaRel=1,amplRel=2.5;
cpmrh(omegaRel);
cpmrh(amplRel);

R omega=omegaRel/rD;
R ampl=amplRel*rD;

R dbr=5*ampl;
R fracbr=0.7;

cpmrh(dbr);
```

```
   cpmrh(fracbr);

   Path pw1=Path::waveLine(pos[2],pos[4],ampl,omega);
   Path pw2=Path::waveLine(pos[2],pos[5],ampl,omega);
   Path pw3=Path::waveLine(pos[2],pos[6],ampl,omega);

   V<Path> vpw1=pw1.brkLine(dbr,fracbr);
   V<Path> vpw2=pw2.brkLine(dbr,fracbr);
   V<Path> vpw3=pw3.brkLine(dbr,fracbr);

   tab2&=vpw1;
   tab2&=vpw2;
   tab2&=vpw3;

   Vec g2Trans=e1*(timeStep*0.5);
   Group g2=g1*Group(g2Trans);
   tab2*=g2;
   for (i=1;i<=ny;++i) pos[i]*=g2;

// creating the table for the final state
   LineArt tab3;
   tab3&=tb3;
   finstate+=sh1;
   LineArt space3=spaceMem;
   space3+=sh2;
   tab3&=finstate;
   tab3&=space3;
// main symbols on table 3
   R xis1=0.6, xis2=0.5;
   cpmrh(xis1);
   cpmrh(xis2);

   Spc pxis(xis1,xis2);
   LineArt dxis=LineArt::disk(pxis,e3,rD);
   lxis.to_(pxis);
   lxis+=namShift;
   lxis+=Vec(0,letterHeight*0.25);
   tab3&=lxis;
   tab3&=dxis;

   Group g3=g2*Group(g2Trans);
   tab3*=g3;
   pxis*=g3;
   lv1is*=g3;
   lv2is*=g3;
   lv3is*=g3;

// creating the axis
   R lArrRel=1.25;
   cpmrh(lArrRel);
   R lArr=timeStep*lArrRel;
   R hArrRel=0.025;
   cpmrh(hArrRel);
   R hArr=lArr*hArrRel;
   R rArrRel=0.25;
   cpmrh(rArrRel);
   R rArr=rArrRel*hArr;
   Spc p1(o);
   Spc p2=p1+e1*lArr;
   LineArt tAxis=LineArt::arr(p1,p2,hArr,rArr);
   R deeper=letterHeight*(1+verticalRelTextShift);
   Spc pText=p2-e1*0.075-e2*deeper;
   Vec shTime=pText-o;
   time+=shTime;
   tAxis&=time;
   Vec shText=-e2*deeper;
   lax1+=shText;

   lax2+=(e1*(timeStep*0.5));
   lax2+=shText;

   lax3+=(e1*timeStep);
   lax3+=shText;
   tAxis&=lax1;
   tAxis&=lax2;
   tAxis&=lax3;

// pathes
   V<Path> ps(6);
   ps[1]=Path(pxi,pos[1]);
   ps[2]=Path(pxi,pos[2]);
   ps[3]=Path(pxi,pos[3]);
   ps[4]=Path(pxis,pos[1]);
   ps[5]=Path(pxis,pos[2]);
   ps[6]=Path(pxis,pos[3]);

   R arrFac1=0.35;
   cpmrh(arrFac1);
   Spc pxi1=pxi+(pos[1]-pxi)*arrFac1;
   Spc pxi2=pxi+(pos[2]-pxi)*arrFac1;
   Spc pxi3=pxi+(pos[3]-pxi)*arrFac1;
   lv1i.to_(pxi1);
   lv2i.to_(pxi2);
   lv3i.to_(pxi3);
   lv1i+=namShift;
   lv2i+=namShift;
   lv3i+=namShift;
   LineArt vi1=LineArt::arr(pxi,pxi1,hArr,rArr);
   LineArt vi2=LineArt::arr(pxi,pxi2,hArr,rArr);
   LineArt vi3=LineArt::arr(pxi,pxi3,hArr,rArr);
```

```
R arrFac2=0.3;
cpmrh(arrFac2);
Spc pxis1=pxis+(pxis-pos[1])*arrFac2;
Spc pxis2=pxis+(pxis-pos[2])*arrFac2;
Spc pxis3=pxis+(pxis-pos[3])*arrFac2;
lv1is.to_(pxis1);
lv2is.to_(pxis2);
lv3is.to_(pxis3);
lv1is+=namShift;
lv2is+=namShift;
lv3is+=namShift;
LineArt vi1s=LineArt::arr(pxis,pxis1,hArr,rArr);
LineArt vi2s=LineArt::arr(pxis,pxis2,hArr,rArr);
LineArt vi3s=LineArt::arr(pxis,pxis3,hArr,rArr);
R rThickRel=0.25;
cpmrh(rThickRel);
R rThick=rD*rThickRel;
vi1.thc_(rThick,nThick);
vi2.thc_(rThick,nThick);
vi3.thc_(rThick,nThick);
vi1s.thc_(rThick,nThick);
vi2s.thc_(rThick,nThick);
vi3s.thc_(rThick,nThick);

LineArt la;

la&=tab1;
la&=tab2;
la&=tab3;
la&=tAxis;

V<Path> pb1=ps[1].brkLine(dbr,fracbr);
V<Path> pb2=ps[2].brkLine(dbr,fracbr);
V<Path> pb3=ps[3].brkLine(dbr,fracbr);
V<Path> pb4=ps[4].brkLine(dbr,fracbr);
V<Path> pb5=ps[5].brkLine(dbr,fracbr);
V<Path> pb6=ps[6].brkLine(dbr,fracbr);

la&=pb1;
la&=pb2;
la&=pb3;
la&=pb4;
la&=pb5;
la&=pb6;

la&=vi1;
la&=vi2;
la&=vi3;
```

```
la&=vi1s;
la&=vi2s;
la&=vi3s;

la&=lv1i;
la&=lv2i;
la&=lv3i;

la&=lv1is;
la&=lv2is;
la&=lv3is;

la.mark(ca);

ca.dis(lineArtProcessing);
cpmwait(tWait,2);
CPM_MZ
```