function [**A**, **Mat**, **G**, **VG**]=deplacement(**M**, **Mat**, **G**, **VG**)

fact=2000; *// constante de la force exercée par une planète sur une autre*

vfact=1000; *// constante de la force exercée par une planète sur le vaisseau*

dt=0.05855; *// pas de temps*

frein=2; *// force de frottement imposée pour que la jouabilité soit meilleure*

nbplanète=length(**M**)/5; *// nombre de planètes*

ax=0; *// accélération horizontale due aux autres objets (somme forces=ma)*

ay=0; *// accélération verticale due aux autres objets*

**A**=**M**;

for j=1:nbplanète do *// pour chaque planète on doit calculer son accélération*

for k=1:nbplanète do *// pour chaque planète son accélération est la résultante de l'attraction des autres planètes*

Capture d’écran du jeu en fonctionnement

if **M**(k,5)==**M**(j,5) then a=-fact; *// coeff négatif si les planètes ont même nature (attraction)*

else a=fact; *// sinon répulsion des planètes*

end

if k<>j then *// attention à ne pas calculer l'action de la planète sur elle-même*

ax(k)=a\*(**M**(k,1)-**M**(j,1))/(((**M**(k,1)-**M**(j,1))\*\*2+(**M**(k,2)-**M**(j,2))\*\*2)\*\*(1.05)); *//cf géométrie*

ay(k)=a\*(**M**(k,2)-**M**(j,2))/(((**M**(k,1)-**M**(j,1))\*\*2+(**M**(k,2)-**M**(j,2))\*\*2)\*\*(1.05)); *// puissance (n+1)/2 dans le cas d'une force en 1/r\*\*n*

else ax(k)=0; ay(k)=0;

end

end

**A**(j,1)=**M**(j,1)+**M**(j,3)\*dt; *//x(t+dt)=x(t)+v(t)dt cf mécanique et DL1(x)*

**A**(j,2)=**M**(j,2)+**M**(j,4)\*dt;

**A**(j,3)=**M**(j,3)+(sum(ax)-frein\***M**(j,3))\*dt; *// v(t+dt)=v(t)+a(t)dt cf mécanique*

**A**(j,4)=**M**(j,4)+(sum(ay)-frein\***M**(j,4))\*dt;

if (**A**(j,1)<0)|(**A**(j,1)>60) then *// si jamais ça sort du cadre par la gauche ou droite la vitesse en x est inversée*

**A**(j,3)=-**A**(j,3); **A**(j,4)=**A**(j,4);

end

if (**A**(j,2)<0)|(**A**(j,2)>60) then *//haut et bas*

**A**(j,3)=**A**(j,3); **A**(j,4)=-**A**(j,4);

end

end

for k=1:nbplanète do *// son accélération est la résultante de l'attraction de toutes les planètes*

if **M**(k,5)==5 then a=-vfact; *// coeff négatif si la nature du vaisseau est 5*

else a=vfact; *// sinon répulsion avec les planètes*

end

ax(k)=a\*(**M**(k,1)-**G**(1))/(((**M**(k,1)-**G**(1))\*\*2+(**M**(k,2)-**G**(2))\*\*2)\*\*(1.005)); *//cf géométrie*

ay(k)=a\*(**M**(k,2)-**G**(2))/(((**M**(k,1)-**G**(1))\*\*2+(**M**(k,2)-**G**(2))\*\*2)\*\*(1.005)); *// puissance (n+1)/2 dans le cas d'une force en r\*\*n*

end

**Mat**(1,:)=**Mat**(1,:)+**VG**(1)\*dt; *//x(t+dt)=x(t)+v(t)dt cf mécanique et DL1(x)*

**G**(1)=**G**(1)+**VG**(1)\*dt;

**Mat**(2,:)=**Mat**(2,:)+**VG**(2)\*dt; *//x(t+dt)=x(t)+v(t)dt cf mécanique et DL1(x)*

**G**(2)=**G**(2)+**VG**(2)\*dt;

**VG**(1)=**VG**(1)+(sum(ax)-frein\***VG**(1))\*dt; *// v(t+dt)=v(t)+a(t)dt cf mécanique*

**VG**(2)=**VG**(2)+(sum(ay)-frein\***VG**(2))\*dt;

endfunction

function colorier(**M**, **N**, **k**, **i**, **couleur**)

for m=1:length(**M**)/5 do xset("color",**M**(m,5)\***k**+**i**) *// i est le choix de la couleur; on colorie les planètes*

xfarc(**M**(m,1)-1,**M**(m,2)+1,2,2,0,23040);

end

if **k**==0 then xset("color",1); xfpoly(**N**(1,:),**N**(2,:)) *//on colorie le vaisseau*

elseif **k**==1 then xset("color",**couleur**); xfpoly(**N**(1,:),**N**(2,:))

end

endfunction

function jauge(**fuel**, **items**, **x**)

xset("color",8);xfpoly([60,57,57,60],[17,17,5,5]); *// traçage de la jauge*

u=[60,57,57,60];v=[17,17,5,5];xpoly(u,v,"lines",1) *// traçage des contours de la jauge*

*//changement de couleur selon le niveau de la jauge (séparation de la jauge en 12 tronçons verticaux)*

if **fuel**>92 then xset("color",13);xfpoly(u,v);

elseif **fuel**>84 & **fuel**<=92 then v(1)=v(1)-1;v(2)=v(1);xset("color",14);xfpoly(u,v);

elseif **fuel**>73 & **fuel**<=84 then v(1)=v(1)-2;v(2)=v(1);xset("color",15);xfpoly(u,v);

elseif **fuel**>65 & **fuel**<=73 then v(1)=v(1)-3;v(2)=v(1);xset("color",3);xfpoly(u,v);

elseif **fuel**>56 & **fuel**<=65 then v(1)=v(1)-4;v(2)=v(1);xset("color",7);xfpoly(u,v);

elseif **fuel**>48 & **fuel**<=56 then v(1)=v(1)-5;v(2)=v(1);xset("color",32);xfpoly(u,v);

elseif **fuel**>40 & **fuel**<=48 then v(1)=v(1)-6;v(2)=v(1);xset("color",26);xfpoly(u,v);

elseif **fuel**>31 & **fuel**<=40 then v(1)=v(1)-7;v(2)=v(1);xset("color",27);xfpoly(u,v);

elseif **fuel**>23 & **fuel**<=31 then v(1)=v(1)-8;v(2)=v(1);xset("color",19);xfpoly(u,v);

elseif **fuel**>15 & **fuel**<=23 then v(1)=v(1)-9;v(2)=v(1);xset("color",20);xfpoly(u,v);

elseif **fuel**>7 & **fuel**<=15 then v(1)=v(1)-10;v(2)=v(1);xset("color",21);xfpoly(u,v);

else v(1)=v(1)-11;v(2)=v(1);xset("color",5);xfpoly(u,v,1);

end

xset("color",0);u=[60,57,57,60];v=[17,17,5,5];xpoly(u,v,"lines",1) *//retrace les contours de la jauge en noir*

if **x**==2 then xset("color",21);xfarc(57,57-3\***items**,2,2,0,23040) ; end *//x=2 si le joueur a attrapé un item, x=1 sinon*

endfunction

function [**Mat**]=rotation(**Mat**, **couleur**, **theta**)

xset("color",1);xfpoly(**Mat**(1,:),**Mat**(2,:)) *//efface le vaisseau précédent*

**Mat**(1,:)=**Mat**(1,:)-G(1);**Mat**(2,:)=**Mat**(2,:)-G(2); *//centre les coordonnées du vaisseau sur l'origine du repère*

**Mat**=[cos(**theta**),-sin(**theta**);sin(**theta**),cos(**theta**)]\***Mat**; *//rotation d'angle theta de centre de rotation G=(0,0)*

**Mat**(1,:)=**Mat**(1,:)+G(1);**Mat**(2,:)=**Mat**(2,:)+G(2); *//rétablit les coordonnées du vaisseau initiales*

xset("color",**couleur**);xfpoly(**Mat**(1,:),**Mat**(2,:)); *// trace le nouveau vaisseau*

endfunction

function [**Mat**, **M**, **G**]=avancer(**Mat**, **M**, **G**, **k**, **couleur**)

*//M est le milieu de BC (A est l’avant du vaisseau)*

*//Mat est la matrice des coordonnes du vaisseau*

*//G est le centre du cercle circonscrit au triangle*

*//k=facteur réglable de variation des coordonnées*

xset("color",1) ; xfpoly(**Mat**(1,:),**Mat**(2,:),1) *//efface le triangle précédent*

*//Séparation de 2 cas : triangle vertical ou quelconque.*

if (**M**(1)-**Mat**(1,2)) <> 0 then *//Cas du triangle quelconque*

**k**=**k**\*(**Mat**(1,2)-**M**(1))/abs(**M**(1)-**Mat**(1,2)); *//k est positif ou négatif selon l'orientation (droite ou gauche) de la pointe du triangle par rapport à M*

a=(**M**(2)-**Mat**(2,2))/(**M**(1)-**Mat**(1,2)); *//a est le coefficient directeur de la droite (AM)*

**Mat**(1,:)=[**Mat**(1,1)+**k**/sqrt(a\*\*2+1),**Mat**(1,2)+**k**/sqrt(a\*\*2+1),**Mat**(1,3)+**k**/sqrt(a\*\*2+1)]; *//modification des abscisses*

**Mat**(2,:)=[**Mat**(2,1)+**k**/sqrt(a\*\*2+1)\*a,**Mat**(2,2)+**k**/sqrt(a\*\*2+1)\*a,**Mat**(2,3)+**k**/sqrt(a\*\*2+1)\*a]; *//modification des ordonnées (dy=a\*dx)*

**G**(1)=**G**(1)+**k**/sqrt(a\*\*2+1); *//modification des abscisses du centre*

**G**(2)=**G**(2)+**k**/sqrt(a\*\*2+1)\*a; *//modification des ordonnées du centre*

else *// Cas du triangle vertical (on ne peut pas calculer a)*

**Mat**(2,:)=**Mat**(2,:)+**k**; *//translate verticalement les ordonnées du vaisseau*

**G**(2)=**G**(2)+**k**; *//translate verticalement les ordonnées du centre du vaisseau*

end

xset("color",**couleur**);xfpoly(**Mat**(1,:),**Mat**(2,:)); *//trace le nouveau triangle*

endfunction

function [**y**]=calculdistance(**G**, **M**)

*//G est le centre du cercle circonscrit au triangle et M est une matrice de coordonnées*

s=size(**M**); l=s(1); *//nombre de lignes de M*

for i=1:l do *//boucle car il peut y avoir plusieurs points*

d(i)=sqrt((**G**(1)-**M**(i,1))\*\*2+(**G**(2)-**M**(i,2))\*\*2); *//norme du vecteur GM*

end

**y**=min(d);

endfunction

function newgame(**pseudo**, **couleur**)

f=figure(25) *//On ouvre le jeu dans une nouvelle fenêtre après avoir appuyé sur le bouton*

a=[0,0];b=[60,60]; *//coordonnées des points extrêmes du cadre de jeu*

plot2d(a,b,axesflag=0), plot2d(b,a,axesflag=0) *// traçage dudit cadre*

xset("color",1); xfpoly([0,0,56,56],[0,56,56,0],1) *//traçage du rectangle noir (zone de jeu)*

*//zone de manœuvre : [2,2,54,54],[2,54,54,2]*

*//constantes :*

c=0.5;*//diminution de fuel a chaque déplacement*

explosion=2.5;*//distance de la planète à partir de laquelle le vaisseau explose*

*//initialisations :*

rep=[0,0,0];fuel=100;items=0;

VG=[0,0]; *//vitesse initiale du vaisseau*

*//initialisation du vaisseau :*

Mat(1,:)=[3.5,4,4.5]; Mat(2,:)=[3,5,3]; *// 1ère ligne : abscisses, 2ème ligne : ordonnées*

G=[4;3.66]; *//centre du cercle circonscrit au triangle*

xset("color",**couleur**);xfpoly(Mat(1,:),Mat(2,:)); *//traçage du triangle*

*//initialisation de la jauge*

jauge(fuel,items,1) *//cf définition de la fonction jauge*

xpoly([60,57,57,60],[17,17,5,5],"lines",1) *//traçage de la jauge*

*//initialisation des planètes*

PLANÈTE=[20,20,0,0,2;30,50,0,0,5;50,10,0,0,5;30,30,0,0,2]; *//traçage des planètes*

xset("color",3); *//définition de la couleur courante comme vert*

*//initialisation de l'item*

item=[rand()\*54+2,rand()\*54+2]; *//coordonnées aléatoires de l'item. On s'arrête à 54 pour que l'item puisse être atteint par le vaisseau dans la zone de jeu*

xfpoly([item(1)-0.5,item(1)+0.5,item(1)+0.5,item(1)-0.5],[item(2)+0.5,item(2)+0.5,item(2)-0.5,item(2)-0.5]) *//traçage de l'item*

while rep(3)<>-1000 do *//tant qu'on ne ferme pas la fenêtre le jeu continue*

tic() *//déclenchement d'un chronomètre qui compte le temps passé sans que l'utilisateur n'effectue d' action*

rep=xgetmouse(); *//on récupère l'action qui a été faite dans la fenêtre (par ex. l'appui sur une touche modifie rep(3))*

temps=0; *//initialisation de la variable temps associée au chronomètre*

if Mat(1,3)<1|Mat(1,3)>54|Mat(2,3)<1|Mat(2,3)>54|Mat(1,1)<1|Mat(1,1)>54|Mat(2,1)<1|Mat(2,1)>54 then time=timer(); *//si le vaisseau est hors du cadre de jeu*

messagebox(["You''re dead "+string(**pseudo**)+"... RIP" "Vous avez recolte "+string(items)+" item(s)"]), rep(3)=-1000;

end

boom=calculdistance(G,PLANÈTE); *//cf fonction calculdistance entre le vaisseau et les planètes*

if boom<explosion then messagebox(["La perturbation gravitationnelle provoquee par la planète vous a fait perdre le controle de votre vaisseau..." " Vous vous ecrasez sur cette terre hostile, peuplee par d''etranges creatures... Au revoir, "+string(pseudo)+" !" "Vous avez recolte "+string(items)+" item(s)"]), rep(3)=-1000;

end

youpi=calculdistance(G,item); *//cf fonction calculdistance entre le vaisseau et l’item*

if youpi<=2 then items=items+1; *//le joueur gagne un item*

xset("color",1);xfpoly([item(1)-0.5,item(1)+0.5,item(1)+0.5,item(1)-0.5],[item(2)+0.5,item(2)+0.5,item(2)-0.5,item(2)-0.5]);*//effaçage de l’item gagné*

xset("color",3); item=[rand()\*54+2,rand()\*54+2]; xfpoly([item(1)-0.5,item(1)+0.5,item(1)+0.5,item(1)-0.5],[item(2)+0.5,item(2)+0.5,item(2)-0.5,item(2)-0.5]); *//traçage d'un nouvel item*

jauge(fuel,items,2) *//cf fonction jauge*

end

if rep(3)==122|rep(3)==115 then *// code ascii : "z"->122 , "s"->115*

temps=toc(); *//fin de l'intervalle de temps pendant lequel les planètes bougent sans appuyer sur aucun bouton*

M=[(Mat(1,1)+Mat(1,3))/2;(Mat(2,1)+Mat(2,3))/2]; *// coordonnées de M le milieu de BC (A en haut)*

if rep(3)==115 then k=-2; *//le vaisseau recule*

else k=2; *//le vaisseau avance*

end

[Mat,M,G]=avancer(Mat,M,G,k,**couleur**); *//cf fonction avancer*

fuel=fuel-c; *//le niveau d'essence diminue de c*

jauge(fuel,items,0) *//cf fonction jauge*

end

*//Rotation*

if rep(3)==100|rep(3)==113 then *//ascii code d <-- 100 et q <-- 113*

temps=toc(); *//fin de l'intervalle de temps pendant lequel les planètes bougent sans appuyer sur aucun bouton*

if rep(3)==100 then *//si on appuie sur d*

theta=-0.3; *// cf fonction rotation*

else theta=0.3;

end

Mat=rotation(Mat,**couleur**,theta); *// cf fonction rotation*

jauge(fuel,items,0) *// cf fonction jauge*

end

*// xpause(5)*

if fuel<=0 then messagebox(["Vous n''avez plus de carburant, "+string(**pseudo**)+"... Vous avez echoue!"," Vous et votre vaisseau allez deriver jusqu''a la fin des temps dans le froid et la solitude de l''univers...", "Vous avez recolte "+string(items)+" item(s)"]), rep(3)=-1000;

end

if temps<>0 then *//si le joueur n'a pas appuyé sur une touche pendant un moment*

colorier(PLANÈTE,Mat,0,1,**couleur**); *//cf fonction colorier*

for i=1:ceil(temps) do [PLANÈTE,Mat,G,VG]=deplacement(PLANÈTE,Mat,G,VG); end

colorier(PLANÈTE,Mat,1,0,**couleur**) *// cf fonction colorier*

xset("color",3);xfpoly([item(1)-0.5,item(1)+0.5,item(1)+0.5,item(1)-0.5],[item(2)+0.5,item(2)+0.5,item(2)-0.5,item(2)-0.5]); *// traçage d'un nouvel item au cas où la planète passe sur l’item et qu'elle l'efface*

end

end *// end de la boucle while*

uicontrol("BackgroundColor",[0.85,0.05,0.05],"String", "Quitter", "Position", [375 10 100, 25], "Callback", "delete(gcf())");

endfunction

function [**y**]=choixpseudo()

**y**=x\_mdialog(['Choix du pseudo'],['Quel est votre nom?'],['Jack'])

endfunction

function [**y**]=choixcouleur()

g=figure(12) ; **y**=getcolor(); close(g);

endfunction