Appendiks B

c-kodar som styrer den aksialbevegelsen til stempela. Koden er baset på kode funne på CFD-forum.

```
#include"udf.h"
DEFINE_GRID_MOTION(motion1,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-dtime*(0.05/2)*(209.4395102)*sin(209.4395102*time); // grid displacement
NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```

```
#include"udf.h"
DEFINE_GRID_MOTION(motion2,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-dtime*(0.05/2)*(209.4395102)*(sin((209.4395102*time)+(6.28315307/7)));
// grid displacement
NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```

```
#include"udf.h"
DEFINE_GRID_MOTION(motion3,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-
dtime*(0.05/2)*(209.4395102)*(sin((209.4395102*time)+((6.28315307*2)/7))); //
grid displacement
NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```

```
#include"udf.h"
DEFINE_GRID_MOTION(motion4,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-
dtime*(0.05/2)*(209.4395102)*(sin((209.4395102*time)+((6.28315307*3)/7))); //
grid displacement
 NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```

```
#include"udf.h"
DEFINE_GRID_MOTION(motion5,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-
dtime*(0.05/2)*(209.4395102)*(sin((209.4395102*time)+((6.28315307*4)/7))); //
grid displacement
NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```

```
#include"udf.h"
DEFINE_GRID_MOTION(motion6,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-
dtime*(0.05/2)*(209.4395102)*(sin((209.4395102*time)+((6.28315307*5)/7))); //
grid displacement
NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```

```
#include"udf.h"
DEFINE_GRID_MOTION(motion7,domain,dt,time,dtime)
{
Thread *tf = DT_THREAD(dt);
face_t f;
Node *v;
real NV_VEC(axis);
real displ;
int n;

displ=-
dtime*(0.05/2)*(209.4395102)*(sin((209.4395102*time)+((6.28315307*6)/7))); //
grid displacement
NV_D(axis, =, 0.0, 0.0, 1.0);

begin_f_loop(f,tf)
{
f_node_loop(f,tf,n)
{
v = F_NODE(f,tf,n);
/* update node if the current node has not been previously
visited when looping through previous faces */
if ( NODE_POS_NEED_UPDATE (v))
{
/* indicate that node position has been update
so that it's not updated more than once */
NODE_POS_UPDATED(v);
NV_V_VS(NODE_COORD(v), =, NODE_COORD(v), +, axis,*,displ);
}
}
}
end_f_loop(f,tf);
}
```