A Appendix

This appendix provides all the details of the constructed model (Fig. 3) in candle format, an exchange format used by Snoopy [11,41]. For more details, please contact, Snoopy HPN user manual [42].

```
colhpn  [Ca3dmodel]
{
constants:
  int D1 = 9;
  int D2 = 9;
  int MID = D1/2;
  int D3 = 9;
  int L = 2;
  int M = 2;
```

```
 int K = 3;
 int N = 2;
 int p = 1;
 int q = 2;
// parameters
 double ks_p = 50;
 double ks_m = 100;
 double km_p = 6;
 double km_m = 0.99;
 double kd_p = 150;
 double kd_m = 300;
 double Di_bs = 0;
 double Di_bm = 20;
 double Di_bd = 15;
 double Di_c = 223;
 double a1 = 81.15;
 double a2 = 1.845;
 double a3 = 200;
 double a4 = 1.15;
 double a5 = 53.9;
 double a0 = 540;
 double b1 = 228;
 double b2 = 0.409;
 double b3 = 188.5;
 double b4 = 0.095;
 double b5 = 4.52;
 double b0 = 80;
 double P = 6320000;
 double Rs = 0.5;
 double C0 = 0.05;
 double PL = 0.04;
 double Pp = 400;
 double Kd = 0.2;
 double Pch = 6320;
 double IP3 = 0.35 ;
 double Di_E = 95;
 double Bs = 400;
 double Bm = 0;
 double dx = 0.5;
 double dy = 0.5;
 double CX0 = 2;
 double CY0 = 2;
 double Cdy = 1;
 double Cdx = 1;
 double k1 = 28.09;
```

```
  double k3 = 9.43;
  double d5 = 1;
  double dz = 0.5;
  double Cdz = 1;
  double CZ0 = 2;
// colorsets:
  Dot = {dot};
  xdim = {1..D1};
  ydim = {1..D2};
  row = {1..L};
  column = {1..M};
  crow = {1..K};
  ccolumn = {1..N};
  zdim = {1..D3};
  c3dim = {1..q};
  dim3 = {1..p};
  Grid3D = PROD(xdim,ydim,zdim);
  Channels = PROD(crow,ccolumn,c3dim);
  Cl = PROD(row,column,dim3);
  ChinCls = PROD(Cl,Channels);
//variables:
  xdim : x;
  ydim : y;
  xdim : a;
  ydim : b;
  row : l;
  column : m;
  crow : k;
  ccolumn : n;
  zdim : z;
  zdim : c;
  dim3 : i;
  c3dim : j;
// colorfunctions:
bool  IsNeighbour(xdim x,ydim y,zdim z,xdim a,ydim b,zdim c)
 {
   (a=x+1 & b=y+1 & c=z+1)|                  //neighbouring conditions
   (a=x-1 & b=y-1 & c=z-1)|
   (a=x   & b=y+1 & c=z+1)|
   (a=x   & b=y+1 & c=z)|
   (a=x+1 & b=y   & c=z+1)|
   (a=x+1 & b=y   & c=z)|
   (a=x+1 & b=y+1 & c=z)|
   (a=x+1 & b=y   & c=z+1)|
   (a=x   & b=y-1 & c=z)|
```

```
   (a=x-1 & b=y   & c=z)|
   (a=x-1 & b=y   & c=z-1) &
   ((1<=a & a<=D1) &(1<=b& b<=D2) & (1<=c& c<=D2))//boundary conditions
 }
//places marking:
//discrete places:
  ChinCls XAct = 0‘all;
  ChinCls X10 = 0‘all;
  ChinCls X00 = 4‘all;
  ChinCls opChans = 0‘all;
  Cl opChCounter = 0‘all;
  ChinCls X01 = 0‘all;
  ChinCls X11 = 0‘all;
//continuous places:
  Grid3D CaE = 700‘(MID,MID,MID);
  Grid3D CaM = 0‘all;
  Grid3D CaS = 0‘all;
  Grid3D CaC = 0‘all;
  Cl clCaC = 0‘all;
// transitions :
// Guard expression
//:transition function;
// stochastic:
 // Ass
    :
    : [XAct + {((l,m,i),(k,n,j))}] & [X10   {((l,m,i),(k,n,j))}]
    : MassAction(a0) ;
// Diss
    :
    : [X10 + {((l,m,i),(k,n,j))}] & [XAct   {((l,m,i),(k,n,j))}]
    : MassAction(b0) ;
// bind_ac
    : [clCaC {(l,m,i)}]
    : [X10 + {((l,m,i),(k,n,j))}] & [X00   {((l,m,i),(k,n,j))}]
    : MassAction((a5*(clCaC+C0))/(1+k1));
// unbind3_ac
    :
    : [X00 + {((l,m,i),(k,n,j))}] & [X10   {((l,m,i),(k,n,j))}]
    : MassAction(b5);
//  bind1_ac
    : [clCaC {(l,m,i)}]
    : [X11 + {((l,m,i),(k,n,j))}] & [X01   {((l,m,i),(k,n,j))}]
    : MassAction((clCaC+C0)*a5/(1+k3));
//  unbind1_ac
    :
```

```
    : [X01 + {((l,m,i),(k,n,j))}] & [X11   {((l,m,i),(k,n,j))}]
    : MassAction(b5);
//  unbind1_ic
    :
    : [X10 + {((l,m,i),(k,n,j))}] & [X11   {((l,m,i),(k,n,j))}]
    : MassAction(b2);
//  unbind_ic
    :
    : [X00 + {((l,m,i),(k,n,j))}] & [X01   {((l,m,i),(k,n,j))}]
    : MassAction((b4*k3+b2)/(1+k3)) ;
//  bind_ic
    : [clCaC {(l,m,i)}]
    : [X01 + {((l,m,i),(k,n,j))}] & [X00   {((l,m,i),(k,n,j))}]
    : MassAction((clCaC+C0)*(a4*k1+a2)/(1+k1));
//  bind1_ic
    : [clCaC {(l,m,i)}]
    : [X11 + {((l,m,i),(k,n,j))}] & [X10   {((l,m,i),(k,n,j))}]
    : MassAction(a2*(clCaC+C0));
// continuous:
//  Leak
    : [CaC {(x,y,z)}]
    : [CaC + {(x,y,z)}] & [CaE   {(x,y,z)}]
    : PL*(CaE CaC) ;
//  DiffE
 {[IsNeighbour(x,y,z,a,b,c)]}
    :
    : [CaE + {(a,b,c)}] & [CaE   {(x,y,z)}]
    :  MassAction(Di_E) ;
 // pump
    :
    : [CaE + {(x,y,z)}] & [CaC   {(x,y,z)}]
    :  (Pp*CaC^2)/(Kd^2+CaC^2);
//  Diffbm
 {[IsNeighbour(x,y,z,a,b,c)]}
    :
    : [CaM + {(a,b,c)}] & [CaM   {(x,y,z)}]
    :  MassAction(Di_ 2bm);
//  kbm_ m
    :
    : [CaC + {(x,y,z)}] & [CaM   {(x,y,z)}]
    :  MassAction(km_ m);
//  kbm_ p
    : [CaM {(x,y,z)}]
    : [CaM + {(x,y,z)}] & [CaC   {(x,y,z)}]
    :  km_ p*(Bm CaM)*CaC;
```

```
//  Kbs_ m
    :
    : [CaC + {(x,y,z)}] & [CaS   {(x,y,z)}]
    :  MassAction(ks_ m );
//  kbs_ p
    : [CaS {(x,y,z)}]
    : [CaS + {(x,y,z)}] & [CaC   {(x,y,z)}]
    :  ks_ p*(Bs CaS)*CaC;
 //  Diffc
 {[IsNeighbour(x,y,z,a,b,c)]}
    :
    : [CaC + {(a,b,c)}] & [CaC   {(x,y,z)}]
    :  MassAction(Di_ c);
//  Diffbs
 {[IsNeighbour(x,y,z,a,b,c)]}
    :
    : [CaS + {(a,b,c)}] & [CaS   {(x,y,z)}]
    :  MassAction(Di_ bs) ;
//  ER2CT
    : [CaC {(x,y,z)}] & [opChCounter {(l,m,i)}]
    : [CaC + {(x,y,z)}] & [CaE   {(x,y,z)}]
    :  lt(sqrt( ([x]*dx (CX0+([l] 1)*Cdx))^2+([y]*dy (CY0+([m] 1)*Cdy))^2 +
       ([z]*dz (CZ0+([i] 1)*Cdz))^2 ),Rs*sqrt( opChCounter))*Pch*(CaE CaC);
//  sum_ Cl_ Ca
    : [CaC {(x,y,z)}]
    : [clCaC + {(l,m,i)}]
    : Rate(CaC ;
//  immediate:
  open
    : [XAct >= {3`((l,m,i),(k,n,j))}] & [opChans < {((l,m,i),(k,n,j))}]
    : [opChans + {((l,m,i),(k,n,j))}] & [opChCounter + {(l,m,i)}]
    : 1 ;
 //  close
    : [XAct < {3`((l,m,i),(k,n,j))}]
    : [opChans   {((l,m,i),(k,n,j))}] & [opChCounter   {(l,m,i)}]
    : 1 ;
```