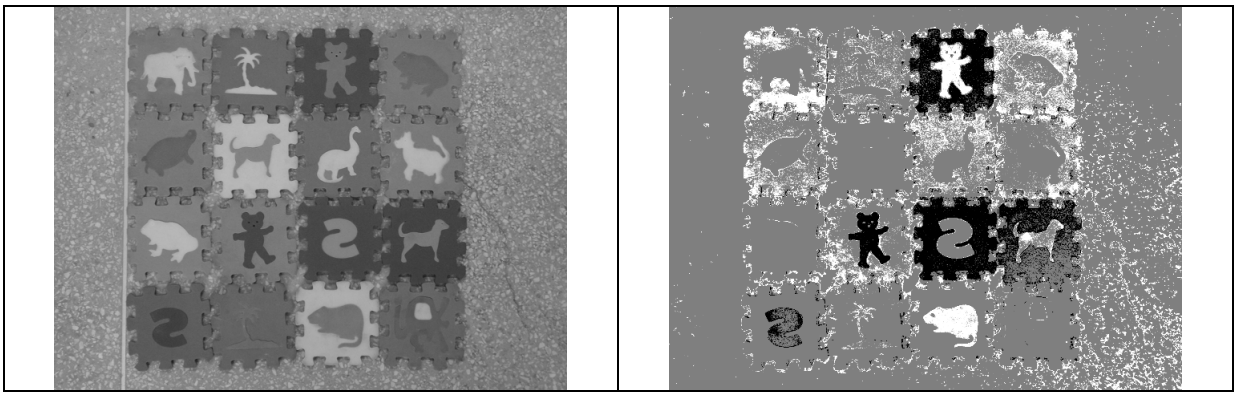
Filtros nebulosos:

```
// Aumenta contraste - pós 2007
#include <proeikon>

//FFN black(-1,0,0,1);
FFN white(0,1,1,2);

void wop(IMGGRY &ent, IMGGRY &sai)
{ ent.backg()=0;
  IMGFLT entf=ent;
  IMGFLT saif(ent.nl(),ent.nc()); saif.backg()=0;

  for (int l=0; l<ent.nl(); ++l)
    for (int c=0; c<ent.nc(); ++c) {
      OUTVAR y; y.v=0;
      fifelse( entf(l,c)==white, 1, y, 1.3, -0.3 );
      saif(l,c)=y.v;
    }

  sai=saif;
}

int main(int argc, char **argv)
{ if (argc!=3) erro("Contras ent.tga sai.tga\n");

  IMGGRY ent;
  le(ent,argv[1]);

  IMGGRY sai;
  wop(ent,sai);

  imp(sai,argv[2]);
}
```

Se o pixel for branco, deixe-o mais branco;
Se o pixel for preto, deixe-o mais preto;

```
// Pinta rato de branco, urso de preto e resto de cinza - pós 2007
// Baseado somente na cor do pixel
#include <proeikon>

double v=0.439;
double d=0.015;

FFN rato(v-2*d,v-d,v+d,v+2*d);

double v2=0.282;
FFN urso(v2-2*d,v2-d,v2+d,v2+2*d);

IMGGRY wop(IMGGRY ent)
{ IMGFLT entf=ent; entf.backg()=0.0;
  IMGFLT saif(ent.nl(),ent.nc()); saif.backg()=0.0;

  for (int l=0; l<ent.nl(); ++l)
    for (int c=0; c<ent.nc(); ++c) {
      OUTVAR y; y.v=0;
      fifelse( entf(l,c)==rato, 1, y, 1.0, 0.5 );
      fifelse( entf(l,c)==urso, 1, y, 0.0, 0.5 );
      saif(l,c)=y.v;
    }

  IMGGRY sai=saif;
  return sai;
}

int main(int argc, char **argv)
{ if (argc!=3) erro("112 ent.tga sai.tga\n");

  IMGGRY ent;
  le(ent,argv[1]);

  IMGGRY sai=wop(ent);

  imp(normaliza(sai),argv[2]);
}
```



```
// Find edge nebuloso - pós 2007
#include <proeikon>

FFN black(-1,0,0,1);
FFN white(0,1,1,2);

void wop(IMGGRY &ent, IMGGRY &sai)
{ ent.backg()=0;
  IMGFLT entf=ent;
  IMGFLT saif(ent.nl(),ent.nc()); saif.backg()=0;

  for (int l=0; l<ent.nl(); ++l)
    for (int c=0; c<ent.nc(); ++c) {

      OUTVAR y;
      double d=fabs(entf(l,c)-entf(l,c+1, 'x'))+
               fabs(entf(l,c)-entf(l+1,c, 'x'))+
               fabs(entf(l,c)-entf(l+1,c+1, 'x'));
      fif( d==black, 1,   y,  1.14 );
      fif( d==white, 5.1, y, -0.14 );

      saif(l,c)=y.v;
    }

  sai=saif;
}

int main(int argc, char **argv)
{ if (argc!=3) erro("FE ent.tga sai.tga\n");

  IMGGRY ent;
  le(ent,argv[1]);

  IMGGRY sai;
  wop(ent,sai);

  imp(sai,argv[2]);
}
```

```
// Detecta arestas
void wop(IMGGRY &ent, IMGGRY &sai)
{ ent.backg()=0;
  IMGFLT entf=ent;
  IMGFLT saif(ent.nl(),ent.nc()); saif.backg()=0;

  for (int l=0; l<ent.nl(); ++l)
    for (int c=0; c<ent.nc(); ++c) {
      OUTVAR y; y.v=0;

      double d=abs(entf(l,c)-entf(l,c+1, 'x'))+
               abs(entf(l,c)-entf(l+1,c, 'x'))+
               abs(entf(l,c)-entf(l+1,c+1, 'x'));

      fif( d==grande, 1, y, 0 );
      fif( d==pequeno, 1, y, 1 );
      saif(l,c)=y.v;
    }

  sai=normaliza(saif);
}
```

Meus artigos sobre filtros nebulosos:

[Ri02] H. Y. Kim, F. A. M. Cipparrone and M. T. C. Andrade, “Technique to Construct Grey-Scale Morphological Operators Using Fuzzy Expert System,” *Electronics Letters,* vol. 33, no. 22, pp. 1859-1861, 1997.

[Cn08] H. Y. Kim, “Filtros Nebulosos no Espaço de Escala,” in *Proc. Simpósio Brasileiro de Telecomunicações*, paper 4140035, 2000