# Some CTRLC-Functions for Manipulation of Simple Figures

Mårtensson, Bengt

1985

*Document Version:*
Publisher's PDF, also known as Version of record

Link to publication

*Citation for published version (APA):*
Mårtensson, B. (1985). *Some CTRLC-Functions for Manipulation of Simple Figures*. (Technical Reports TFRT-7295). Department of Automatic Control, Lund Institute of Technology (LTH).

*Total number of authors:*
1

CODEN:LUTFD2/(TFRT-7295)/1-5/(1985)

# Some CTRLC-functions for Manipulation of Simple Figures

Bengt Mårtensson

Department of Automatic Control
Lund Institute of Technology
October 1985

**Department of Automatic Control**
**Lund Institute of Technology**
P.O. Box 118
S-221 00 Lund Sweden

*Document name*
Report

*Date of issue*
October 15, 1985

*Document Number*
CODEN: LUTFD2/(TFRT-7295)/1-5/(1985)

*Author(s)*
Bengt Mårtensson

*Supervisor*

*Sponsoring organisation*

*Title and subtitle*
Some CTRLC-functions for Manipulation of Simple Figures

*Abstract*

A few CTRLC-functions for image manipulation according to [Nielsen] and [Mårtensson] has been written. It is believed that they might be of use also for other purposes.

These functions are

| | |
|---|---|
| `sigma` | Draws concentric regular $n$-gons with prescribed radii |
| `e` | Embedding of the marking plane in 3-dimensional space |
| `mfunc` | Action of the Euclidean group on points in 3-space |
| `ifunc` | Image projection of points in 3-space to the image plane |
| `pcoord` | Plot routine |
| `ppim` | The composition of all the above functions |
| `pp` | A call to `ppim` using global variables |


*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
English

*Number of pages*
5

*Security classification*

*Recipient's notes*

# Some CTRLC-functions for Manipulation of Simple Figures

Bengt Mårtensson
85-10-15

A few CTRLC-functions for image manipulation according to [Nielsen] and [Mårtensson] has been written. It is believed that they might be of use also for other purposes.

These functions are

| | |
|---|---|
| `sigma` | Draws concentric regular $n$-gons with prescribed radii |
| `e` | Embedding of the marking plane in 3-dimensional space |
| `mfunc` | Action of the Euclidean group on point in 3-space |
| `ifunc` | Image projection of points in 3-space to the image plane |
| `pcoord` | Plot routine |
| `ppim` | The composition of all the above functions |
| `pp` | A call to `ppim` using global variables |

The first 6 of these are available in the CTRLC-library `image.lib`. The last is not a function in CTRLC-sense, but is intended to be run by a `do pp`.

The name of the functions are essentialy the same as in [Nielsen] and [Mårtensson]. To avoid name collision, the funtions $e$ and $M$ are called `efunc` and `mfunc` respectively. Note that it is the augmented version, acting on a set of vectors, ($e^m$, $M_\xi^m$, $i^m$ in the terminology of [Mårtensson]) that has been implemented.

All functions acting on $m$ vectors in $r$-space requires the coordinates of the input to be in the form of a $r \times m$ matrix.

Figures generated by these functions are shown in Figure 1.

The rest of this manual is devoted to a slightly more detailed description of the functions and their arguments.

### sigma

Call: `xmym = sigma(n,k)`, where $n = 3, 4, 5, \ldots$, and $k = (k_1, \ldots, k_r)$. The number $r$ is not fixed. This will assign to `xmym` the coordinates of the vertices of $r$ concentric regular $n$-goms, the $i$-th one inscribed in a circle of radius $k_i$. For compatibility with the plotting routines, the largest is suggested to be put equal to 1.

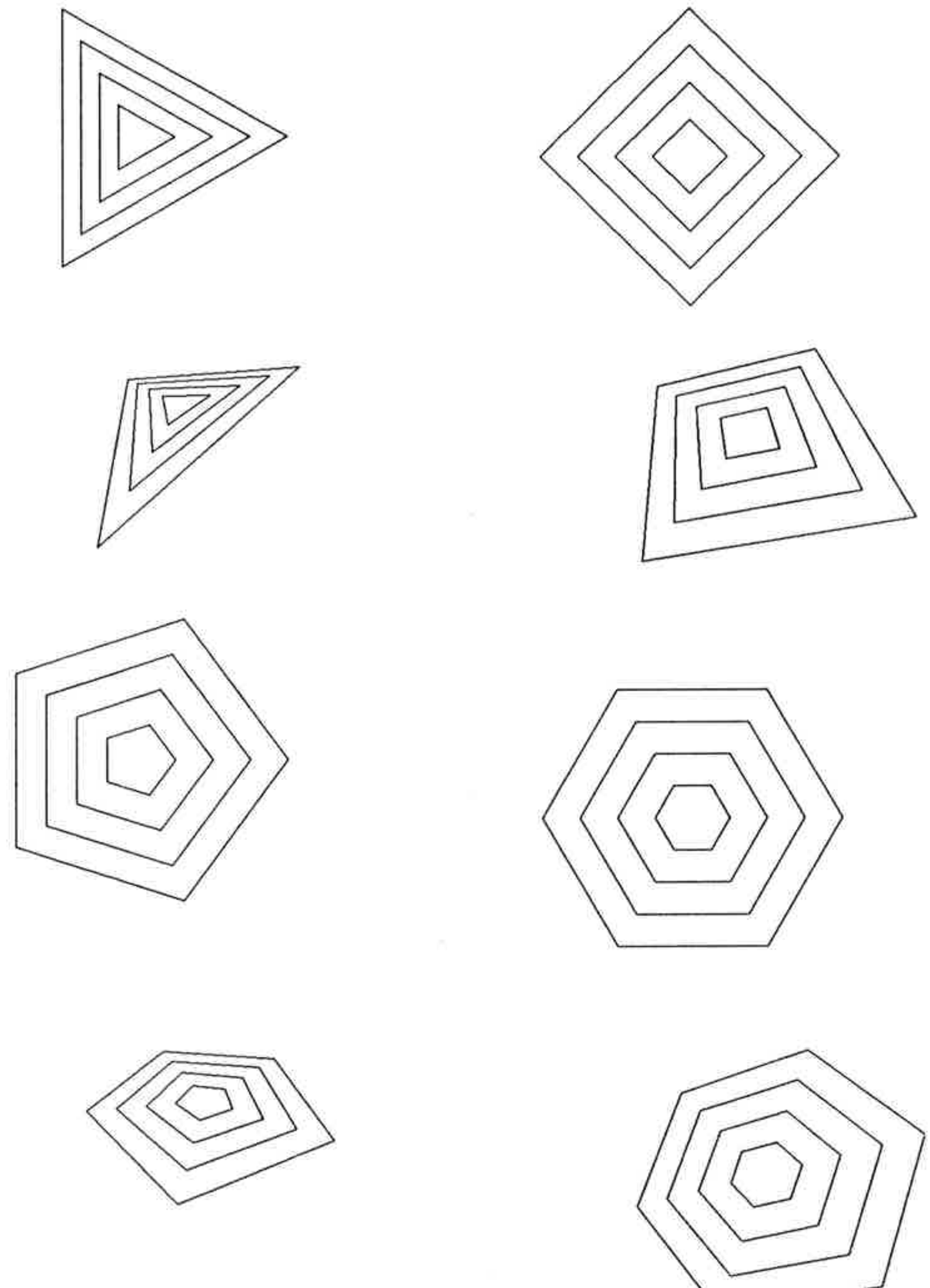

*Figure 1. Some perspective invariant markings based on regular, concentric n-gons. Original image and image after transformation shown.*

**efunc**

Call: `[xyz] = e(xy)`. This is simply the inbedding of the two-dimensional (marking-) plane into $\mathbb{R}^3$, as given by $(x, y)^T \longrightarrow (x, y, 0)^T$, and the multi vector generalization.

**mfunc**
Call: [xcyczc] = mfunc(xi,xyz). Here xyz is a $3 \times r$ matrix of coordinates of points in $\mathbb{R}^3$, and xi = $\xi = (x, y, z, \theta, \phi, \psi)^T$, the latter beeing coordinates on (an open dense subset of) the Euclidean group, viz. a translation vector $(x, y, z)$, and $SO(3)$ parametrized by the Euler angles.

**ifunc**
Call: [xiyi] = ifunc(f,xyz). This is the perspective projection of the points with coordinates in the matrix xyz to the plane $\{z = 0\}$. The optical center is placed in $(0, 0, -f)^T$.

**pcoord**
Call: [] = pcoord(m,coord). This function plots the points of the matrix coord as if they where the vertices of (not necessarily regular or concentric) $m$-gons. The height- and length-scales are adjusted to be equal. To be fully inside the screen, no coordinate should be of magnitude larger than 1.

**ppim**
Call: [] = ppim(m,k,xi,f). This is the composition of all the functions described above. The variables has the same meaning as before.

**pp**
Call: do pp. This is not a funtion, and it is not included in the library image.lib. It makes a call to ppim, and requires that the variables in the call of ppim has been set up as global CTRLC-variables, under these names.

**References**

Mårtensson, B. (1985) "*Perspective Invariant Markings - A Geometric Approach*", CODEN:LUTFRT/(TFRT-7294), Department of Automatic Control, Lund Institute of Technology

Nielsen, L. (1985) "*Simplifications in Visual Servoing*", PhD Thesis, CODEN:LUTFRT/(TFRT-1027), Department of Automatic Control, Lund Institute of Technology


## Appendix. The program code

---

*sigma*

```
//[xmym] = sigma(n,k);
x = 1;
y = 1;
for i = 1:max(size(k)),...
  for j = 0:n-1,...
    x = [x,k(i)*cos(2*pi*j/n)];..
```

```
    y = [y,k(i)*sin(2*pi*j/n)];..
  end,..
end,
xmym = [x(:,2:(n*max(size(k))+1));y(:,2:(n*max(size(k))+1))];
```

---

*e*

```
//[xyz] = e(xy);
xyz = [xy;0*ones(1,max(size(xy)))];
```

---

*mfunc*

```
//[xcyczc] = mfunc(xi,xyz);
// Transforms the columns in xyz under the Euclidean group
// parametrized by xi.
r = xi(1:3);
theta = xi(4);
fi = xi(5);
psi = xi(6);
Q1 = [cos(theta) -sin(theta)         0;
      sin(theta)  cos(theta)         0;
         0            0              1];
Q2 = [   1            0              0;
         0         cos(fi)       -sin(fi);
         0         sin(fi)        cos(fi)];
Q3 = [cos(psi) -sin(psi)             0;
      sin(psi)  cos(psi)             0;
         0            0              1];
Q = Q1*Q2*Q3;
xcyczc = Q*xyz + r*ones(1,max(size(xyz)));
```

---

*ifunc*

```
//[xiyi] = ifunc(f,xyz);
//Projects the columns in xyz to the plane {z = 0}
//under image projection with optical center in (0,0,-f)'
frow = f*ones(1,max(size(xyz)));
scaler = diag(frow ./ (xyz(3,:) + frow));
xiyi = xyz(1:2,:)*scaler;
```

---

*pcoord*

```
//[] = pcoord(m,coor);
erase;
//Set up proper relation between horizontal and vertical scale
w = [149/190 1;0 0];
window(w);
//Transform the [-1,1]-interval to [0,1]
```

```
coor = 0.5*(coor + ones(coor));
for i = 1:max(size(coor))/m,..
  x = coor(1,(i-1)*m+1:i*m);..
  y = coor(2,(i-1)*m+1:i*m);..
  x = [x,x(1)];..
  y = [y,y(1)];..
  pline(x,y);..
end,
```

---

## *ppim*

```
//[] = ppim(m,k,xi,f);
pcoord(m,ifunc(f,mfunc(xi,e(sigma(m,k)))));
```

---

## *pp*

```
// pp
ppim(m,k,xi,f);
```