

Anhang

ANHANG A : BERECHNUNG DER SCHALTFLÄCHEN	A-1
Schaltfläche des einfachen Fuzzy-Reglers	A-1
Schaltfläche des Fuzzy-Reglers mit Drehzahlvorsteuerung	A-2
Programmlistings:	A-4
F_DEF	A-4
F_INI	A-7
F_INTPY	A-7
F_MINPRD	A-8
F_REGBAS	A-9
ANHANG B : GRUNDLAGEN DER FUZZY-LOGIC	B-12
Fuzzyfizierung	B-13
Inferenz	B-14
Operatoren	B-14
Anwendung auf regelbasierte Systeme	B-15
Defuzzyfizierung	B-16
MOM (MEAN OF MAXIMA) - Methode	B-17
COA (Centre of area) - Methode	B-17
ANHANG C : S-FUNCTIONS	C-20
Meßwertaufbereitung	C-20
Drehzahlvorsteuerung	C-23
Sollwertgenerator	C-25
ANHANG D : ANIMATIONSPROGRAMM	D-26
Datei-Menu	D-26
Animation-Menu	D-26
Parameter-Menu	D-27

Anhang A : Berechnung der Schaltflächen

Schaltfläche des einfachen Fuzzy-Reglers

Schaltfläche des Fuzzy-Reglers mit Drehzahlvorsteuerung

Ein Regelzyklus mit Fuzzyfizierung, Inferenz und Defuzzyfizierung braucht Zeit. Wenn man diese Rechenoperationen nicht zur Laufzeit des Reglers sondern vorher durchführt, erreicht man erhebliche Geschwindigkeitsverbesserungen.

Aus diesem Grund wird aus den Fuzzy-Sets und den Regeln eine Schaltfläche berechnet, die in Form einer zweidimensionalen Matrix abgespeichert wird. Der Fuzzy-Algorithmus wird für eine Anzahl von Wertepaaren „durchgespielt“ und die Ergebnisse abgespeichert. Zur Laufzeit werden Zwischenwerte durch Interpolation gewonnen.

Die Schaltflächen der beiden Fuzzy-Regler sind auf den vorhergehenden Seiten abgedruckt

Zur Berechnung wurde auf Unterprogramme zurückgegriffen, die bereits im Rahmen einer Diplomarbeit [5] entwickelt wurden. Diese Routinen wurden leicht modifiziert und angepasst. Die Grundstruktur blieb allerdings unverändert.

Im folgenden ein Überblick über die Einzelroutinen:

F_DEF

enthält die Definitionen der Fuzzysets für Regeldifferenz, Drehwinkel und Stellgröße.

F_INI

ist das Hauptprogramm, das alle anderen Unterprogramme aufruft.

F_INTPY

ist eine Interpolationsroutine für F_MINPRD.

F_MINPRD

führt die Fuzzyfizierung, Inferenz und Defuzzifizierung durch. Dabei wird die COA-Methode mit doppelter Berücksichtigung der Schnittflächen verwendet.

F_REGBAS

enthält die linguistischen Regeln in Form einer Matrixstruktur.

Die Programmlistings entstammen von dem einfachen Fuzzy-Regler. Bei dem Regler mit Drehzahlvorsteuerung unterscheidet sich lediglich die Datei „F_DEF“, die andere Werte für die Definition der Fuzzy-Sets der Stellgröße enthält.

Programmlistings:**F_DEF**

```

function [Sx,Sy,Nx,Ny,ventx,venty] = f_def
%-----
%
%           FUZZY CONTROL
%
%           Datendefinitionen
%
%           Unterprogramm zum Simulationsprogramm f_ini.m
%           Übergabe von Ordinatenwerten und Abzissenwerten
%           der Regeldifferenz, Drehwinkel, Drehzahl
%-----
%-----          Regeldifferenz          -----
nLS= [-0.5  1      % Negativ Groß
      -0.08  1
      -0.08  1
      -0.05  0];

nmS= [-0.08  0      % Negativ Klein
      -0.05  1
      -0.05  1
      0      0];

nsS= [-0.05  0      % Zero
      0      1
      0      1
      0.05  0];

zrS= [ 0      0      % Positiv Klein
      0.05  1
      0.05  1
      0.08  0];

psS= [ 0.05  0      % Positiv Mittel
      0.08  1
      0.08  1
      0.12  0];

pmS= [ 0.08  0      % Positiv groß
      0.12  1
      0.12  1
      0.5   1];

Sx= [nLS(:,1), nmS(:,1), nsS(:,1), zrS(:,1), psS(:,1), pmS(:,1)];
Sy= [nLS(:,2), nmS(:,2), nsS(:,2), zrS(:,2), psS(:,2), pmS(:,2)];

%-----          Drehwinkel fi          -----
nmN= [ 0      1      % NL1
      0      1
      0      1
      pi/4  0];

nsN= [ 0      0      % L1
      pi/4  1
      2.064 1
      2.064 0];

zrN= [ 2.064 0      % L2
      2.064 1
      3*pi/4  1

```

```
    pi    0 ];  
psN= [3*pi/4 0          % NL 2  
      pi    1  
      pi    1  
      5*pi/4 0];  
pmN= [ pi    0          % L3  
      5*pi/4 1  
      7*pi/4 1  
      2*pi   0];  
pLN= [7*pi/4 0          % NL 3  
      2*pi   1  
      2*pi   1  
      2*pi   1];  
Nx= [nmN(:,1), nsN(:,1), zrN(:,1), psN(:,1), pmN(:,1), pLN(:,1)];  
Ny= [nmN(:,2), nsN(:,2), zrN(:,2), psN(:,2), pmN(:,2), pLN(:,2)];
```

```

%----- Stellgröße -----
nhh=[ 0    1;      % Negativ Groß
      0    1;
      0    1;
      0.6  0];

nLh=[ 0    0;      % Negativ Klein
      0.6  1;
      0.6  1;
      2.23 0];

nmh=[ 0.6  0;      % Zero
      2.23  1;
      2.23  1;
      4     0];

nsh=[ 2.23  0;      % Positiv klein
      4     1;
      4     1;
      6     0];

zrh=[ 4     0;      % Positiv mittel
      6     1;
      6     1;
      8     0];

psh=[ 6     0;      % Positiv Groß
      8     1;
      8     1;
      10    0];

pmh=[ 8     0;      % Positiv sehr Groß
      10    1;
      10    1;
      11    0];

pLh=[ 0     0;
      0     0;
      0     0;
      0     0];

phh=[ 0     0;
      0     0;
      0     0;
      0     0];

%----- Ausgang mit 9 Fuzzy Sets -----

ventx=[nhh(:,1),nLh(:,1),nmh(:,1),nsh(:,1),zrh(:,1),psh(:,1),pmh(:,1),pLh
(:,1),phh(:,1)];
venty=[nhh(:,2),nLh(:,2),nmh(:,2),nsh(:,2),zrh(:,2),psh(:,2),pmh(:,2),pLh
(:,2),phh(:,2)];

```

F_INI

```

-----
%
%
%           Fuzzy - Regelung eines Pleuelgetriebes
%
%           Hauptprogramm
%           dieses Programm wird als eine Art
%           Initialisierungsprogramm vor der ein-
%           gentlichen Simulation aufgerufen
%
%           Sfaktor, nfaktor sind die Skalierungsfaktoren des Fuzzy-
%           Regler-Eingangs
%
%           sstat, sddreh sind die als Vektoren definierten
%           Eingangsgrößen zur Generierung der Schaltgeraden
%
%           crisp = Schaltgerade in Matrixform des Fuzzy-Reglers
%
-----
[Sx,Sy,Nx,Ny,ventx,venty] = f_def;           %--- Datendefinition

stat= [-0.5:.05:-0.2, -0.15:.03:0.15, 0.2:.05:0.5 ];
ddreh=[0:.2:2*pi];

crisp= f_minprd(Sx,Sy,Nx,Ny,ventx,venty,stat,ddreh); %-- Min-Prod Methode

```

F_INTPY

```

function y = intpy(xmat,ymat,x0)

%
%-----
%           Interpolationsroutine
%           Input:  x-Wert
%           Output: y-Wert
%
%-----

[m,n]=size(xmat);

y= zeros(n,1);
for i= 1:n
    kk = min(find(xmat(:,i) >= x0));
    if kk==[]
        y(i)=ymat(m,i);
    elseif kk==1
        y(i)=ymat(1,i);
    else
        alpha = ( xmat(kk,i)-x0 )/( xmat(kk,i)-xmat(kk-1,i) );
        y(i) = alpha*ymat(kk-1,i) + (1-alpha)*ymat(kk,i);
    end
end
end

```

F_MINPRD

```

function crisp= f_minprd(Sx,Sy,Nx,Ny,ventx,venty,stat,ddreh)
%-----
%
%           Minimum - Produkt - Methode
%
%           Unterfunktion von: f_ini.m
%
%           Berechnung der Fuzzifizierung und Defuzzifizierung
%
%           übergebene Variablen:
%           Sx,Sy : Fuzzy - Sets der Regeldifferenz
%           Nx,Ny : Fuzzy - Sets des Drehwinkels
%           ventx,venty : Fuzzy - Sets der Stellgröße
%           stat, ddreh : Eingangsvektoren zur Generierung
%                       der Schaltgeraden
%
%           zurückgegebene Variable:
%           crisp : Schaltgerade in Matrixform
%-----

for i=1:length(stat)
    for j=1:length(ddreh)

        [m,n]= size(Sx);
% Fuzzifizierung
        fuzyS= f_intpy(Sx(:,1:n),Sy(:,1:n),stat(i));
% Regeldifferenz fuzzifiziert y
        Sfy= [Sy(1,:) ; fuzyS' ; fuzyS' ; Sy(m,:)];
% Regeldifferenz fuzzifiziert x
        Sfx= Sx;

% analog Drehzahl
        [m,n]= size(Nx);
% eigentliche fuzzifizierte Größe
        fuzyN=f_intpy(Nx(:,1:n),Ny(:,1:n),ddreh(j));
% nur fuer bildliche Darstellung
        Nfy= [Ny(1,:) ; fuzyN' ; fuzyN' ; Ny(m,:)];
        Nfx= Nx;
% Durchnudeln der Regelbasis

        fuzyh = f_regbas(fuzyS,fuzyN);      % ----- Regelbasis

% fuzzifizierten Ausgangsgroößen
        [m,n]= size(venty);
        hfy= [venty(1,1:n); fuzyh' ; fuzyh' ; venty(m,1:n)];
    end
end

```

```

%-----
%
%               Defuzzifizierung der Ausgangsgröße
%               durch die
%               COA - Methode
%
%   Variablen:
%   ventx : Punkte auf der Abzisse
%   hfy   : die zugehörigen Fuzzy Ordinatenwerte
%   crisp : welches den auf die Abzisse transformierten
%           Flächenschwerpunkt angibt
%-----
n= size(ventx,1);
z= ( ventx(2:n,:)-ventx(1:n-1,:) ).*...
    ( ( 2*ventx(2:n,:)+ventx(1:n-1,:) ).*...
      hfy(2:n,:)+ ( 2*ventx(1:n-1,:)+ventx(2:n,:) ).*hfy(1:n-1,:) );
nen=3*( ventx(2:n,:)-ventx(1:n-1,:) ).*( hfy(1:n-1,:)+hfy(2:n,:) );
z= sum(sum(z)');
nen= sum(sum(nen)');
if nen== 0
    nen= 1;
end
crisp(i,j)= z/nen; % defuzzifizierte Wert
end
end
% crisp ist die fertige fuzzy matrix die in der simulink funktion
auftaucht

```

F_REGBAS

```

function fuzyh= f_regbas(fuzyS,fuzyN)
%-----
%
%               Unterprogramm zur Erstellung der linguistischen Regeln
%
%               Übergabe der Ordinatenwerte der einzelnen
%               Zugehörigkeitsfunktionen fuzyS, fuzyN
%
%               zurückgegebene Variable : fuzyh
%               der Form:
%               fuzyh = [nh nL nm ns zr ps pm pL ph]'
%-----

fuzyS= [fuzyS(6);fuzyS(5);fuzyS(4);fuzyS(3);fuzyS(2);fuzyS(1)];
p=size(fuzyS,1);
rm= zeros(p,p);

% Hier werden alle Zugehörigkeiten der Regeln generiert und zwar
% spaltenweise, d.h in einem
% Schritt werden hier 6 Regeln abgearbeitet. Also z.B pm mit nm ns zr ps
% pm und pl.

for i=1:p
    rm(:,i)= min( fuzyS(i),fuzyN );
end

% Nur mit Zahlen, noch nicht Zugehörigkeitsfunktionen gefüllte Matrix

```

```

% hier erfolgt die Zuordnung der Zahlen mit fuzzy sets
%
%          |          Regeldifferenz
%          |          pg      pm      pk      zr      nk      ng
%          |-----|-----|-----|-----|-----|-----|
regel = [ 'pgg','pg ','pm ','pm ','zr ','nk ' ;          % NL1
%          | 'pg ','pm ','pm ','zr ','nk ','ng ' ;          % L1
%          | 'pg ','pm ','pk ','zr ','nk ','ng ' ;          % L2
%          | 'pgg','pg ','pm ','pk ','zr ','nk ' ;          % NL2
%          | 'pg ','pm ','pm ','pm ','zr ','nk ' ;          % L3
%          | 'pgg','pg ','pm ','pm ','zr ','nk '];          % NL3
%
%
%          pgg
%
pgg=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='p')*(regel(i,j*3-1)=='g')*(regel(i,j*3)=='g'),
            pgg = max(pgg,rm(i,j));
        end
    end
end
%
%          pg
%
pg=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='p')*(regel(i,j*3-1)=='g')*(regel(i,j*3)==' '),
            pg = max(pg,rm(i,j));
        end
    end
end
%
%          pm
%
pm=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='p')*(regel(i,j*3-1)=='m')*(regel(i,j*3)==' '),
            pm = max(pm,rm(i,j));
        end
    end
end
%
%          pk
%
pk=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='p')*(regel(i,j*3-1)=='k')*(regel(i,j*3)==' '),
            pk = max(pk,rm(i,j));
        end
    end
end
%
%          zr
%
zr=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='z')*(regel(i,j*3-1)=='r')*(regel(i,j*3)==' '),
            zr = max(zr,rm(i,j));
        end
    end
end

```

```
end
end
%
% nk
%
nk=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='n')*(regel(i,j*3-1)=='k')*(regel(i,j*3)==' '),
            nk = max(nk,rm(i,j));
        end
    end
end
%
% ng
%
ng=0;
for i = 1 : 5,
    for j = 1 : 6,
        if (regel(i,j*3-2)=='n')*(regel(i,j*3-1)=='g')*(regel(i,j*3)==' '),
            ng = max(ng,rm(i,j));
        end
    end
end
%
% Ausgang = (ordinate) Werte der fuzzy sets
%
fuzzyh= [ng, nk, zr, pk, pm, pg, pgg, 0, 0]';
```

Anhang B : Grundlagen der Fuzzy-Logic

In unserem alltäglichen Umgang sind wir es gewohnt mit unscharfen Aussagen umzugehen. Wir sagen z.B. „Der Mann ist groß.“, ohne uns Gedanken zu machen was der Begriff „groß“ wirklich über die Körpergröße aussagt. Trifft man diese Aussage im Kreise von Basketballspielern, hat sie eine andere physikalische Bedeutung als in einer Gruppe von Kindern.

Die Fuzzy-Logic ist ein Hilfsmittel zur Handhabung von unscharfen Informationen [12]. Ihr Hauptvorteil besteht darin, daß Prozesse nicht mathematisch und physikalisch exakt beschrieben werden brauchen. Es reichen heuristische Kenntnisse, die auf der Erfahrung beruhen, aus.

Diese können dann in einfachen Regeln formuliert werden.

WENN die Kurbel im nichtlinearen Bereich ist, DANN soll sie schneller drehen.

Meßwerte werden in der Praxis aber exakt erfaßt, z.B. Schlittenposition $x(t) = 0,534$ m. Ebenso müssen Prozesse durch genaue Stellgrößen gesteuert werden. Es reicht nicht aus dem Motor zu sagen, daß er „schnell“ drehen soll. Hier ist die Angabe einer Steuerspannung notwendig.

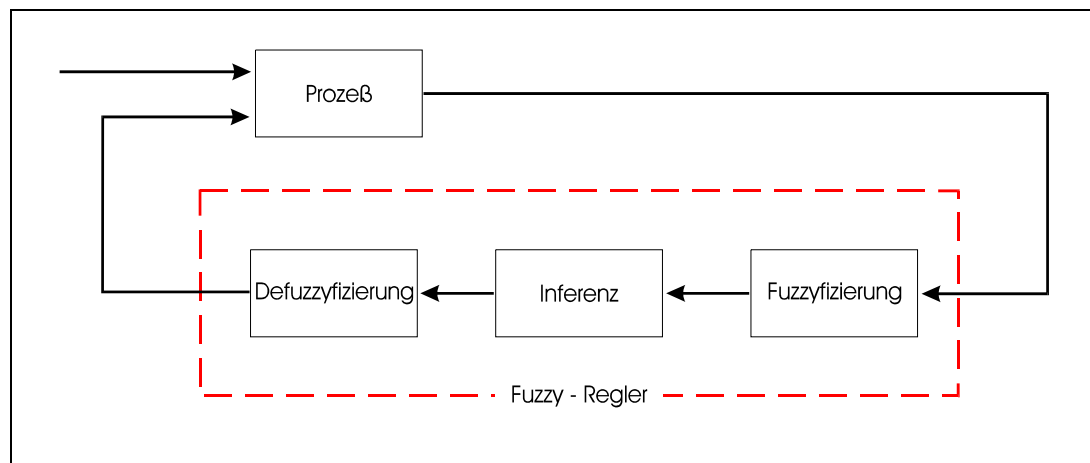


Abbildung Anhang B -1 [5]: Regelkreis mit Fuzzy-Regler

Abbildung Anhang B -1 stellt dar, wie exakte Größen in einem Fuzzy-Regler verarbeitet werden.

Der Meßwert wird zuerst in eine unscharfe Beschreibungsform umgewandelt. Diesen Vorgang nennt man **Fuzzyfizierung**. Auf diese Aussagen werden Regeln angewendet (**Inferenz**). Das Resultat ist eine unscharfe Ausgangsgröße, die dann bei der **Defuzzyfizierung** wieder in eine genaue Stellgröße umgewandelt wird. Die exakten Größen nennt man „Crisp-Werte“.

Fuzzyfizierung

Zur unscharfen Beschreibung von Prozessen verwendet man „linguistische Variable“. Man weist z.B. der Regelabweichung Begriffe zu: „positiv groß“, „negativ klein“, „null“ usw. Die Menge der Werte, die einem Begriff zugeordnet werden können, nennt man Fuzzy-Set.

Aufgabe der Fuzzyfizierung ist es einen Crisp-Wert zu einer oder mehreren Fuzzy-Mengen zuzuordnen. Dies geschieht über die sog. „Zuordnungsfunktion μ “, die jedem scharfen Wert die Zugehörigkeit zu den Fuzzy-Mengen zuordnet. μ nimmt dabei Werte zwischen 0 und 1 an.

Abbildung Anhang B -2 und Abbildung Anhang B -3 zeigen die Zugehörigkeitsfunktionen für den Kurbelwinkel und die Regelabweichung. Bei der Definition bleiben vier Freiheitsgrade:

- Anzahl der Zugehörigkeitsfunktionen
- Form der Zugehörigkeitsfunktion
- Lage der Zugehörigkeitsfunktion
- Breite der Zugehörigkeitsfunktion

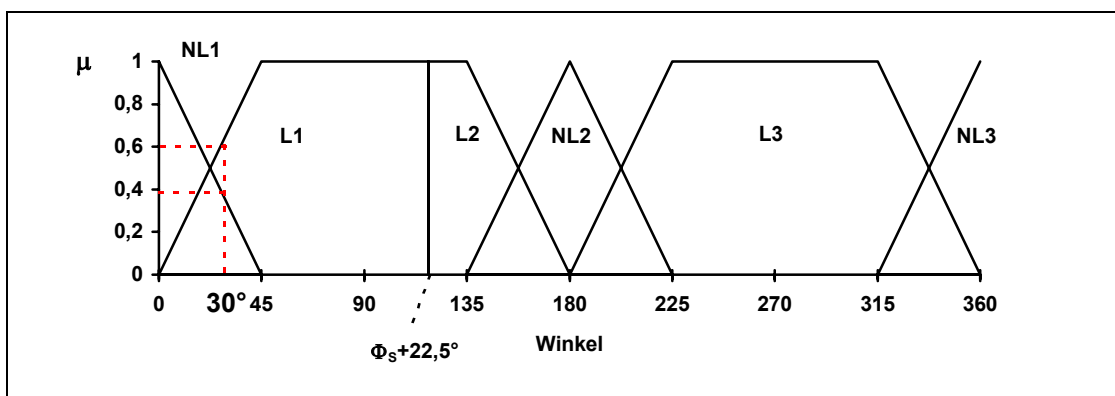


Abbildung Anhang B -2: Fuzzy-Sets für die nichtlinearen Bereiche

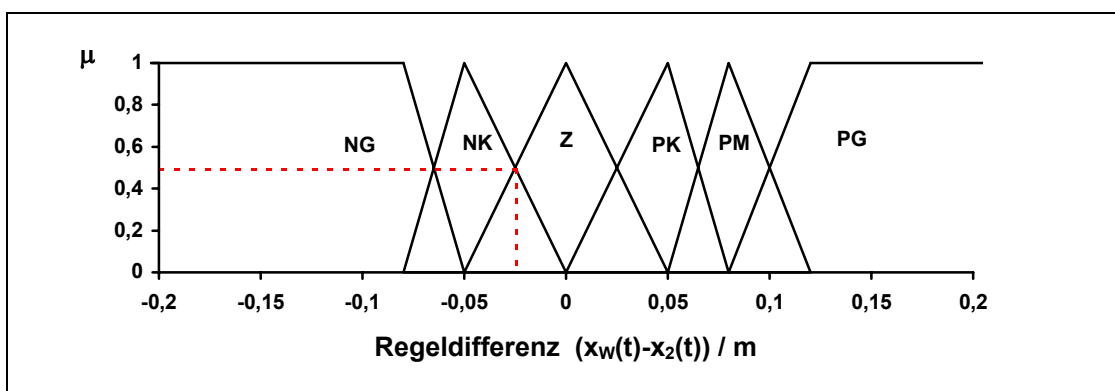


Abbildung Anhang B -3: Fuzzy-Sets für die Regeldifferenz

Beispiel:

Der Regelabweichung $x_w(t) - x_2(t) = 0,025$ m und dem Kurbelwinkel $\Phi(t) = 30^\circ$ sollen linguistische Variable zugeordnet werden. Die Zugehörigkeit kann aus der Definition der Fuzzy-Sets abgelesen werden (gestrichelte Linie).

Für die Regelabweichung gilt:

$$\begin{aligned} \mu &= 0,5 & \text{NK (negativ klein)} \\ \mu &= 0,5 & \text{Z (null)} \end{aligned}$$

Für den Kurbelwinkel gilt:

$$\begin{aligned} \mu &= 0,4 & \text{NL1 (nichtlinearer Bereich 1)} \\ \mu &= 0,6 & \text{L1 (linearer Bereich 1)} \end{aligned}$$

Für die anderen Fuzzy-Mengen ist die Zugehörigkeit jeweils $\mu = 0$.

Inferenz**Operatoren**

Wir sind es gewohnt Aussagen entweder mit „und“ oder „oder“ zu verknüpfen. Wir sagen z.B. „Wenn das Auto schnell ist und die Straße glatt ist, dann kann ein Unfall passieren.“

Diese Operatoren werden auch für die Anwendung auf Fuzzy-Sets definiert:

UND ist der Durchschnitt der Flächen unter dem Graphen ihrer Zugehörigkeitsfunktion.

ODER ist die Vereinigung der Flächen unter den Graphen ihrer Zugehörigkeitsfunktion.

In Abbildung Anhang B -4 werden die Aussagen

NG (negativ groß) UND NK (negativ klein) und
PK (positiv groß) UND PM (positiv mittel)

veranschaulicht.

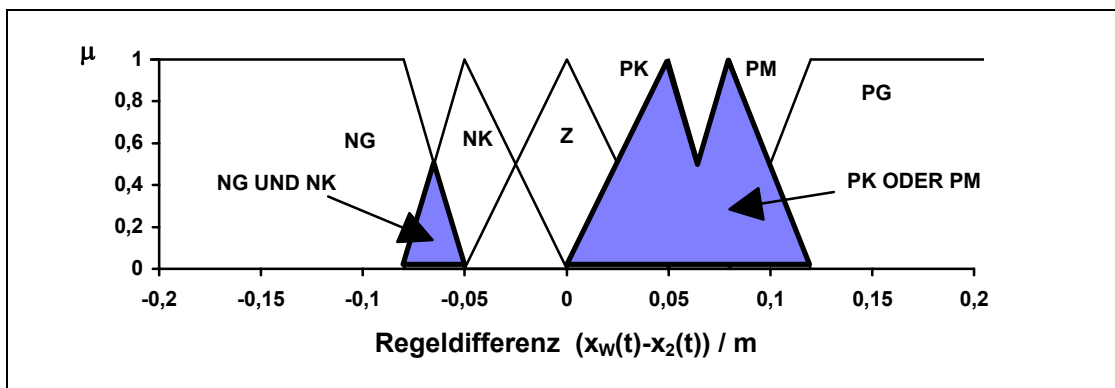


Abbildung Anhang B -4: Beispiel für den UND-Operator und den ODER-Operator

Es resultieren die Vereinbarungen:

- UND-Verknüpfung → MIN-Operator
- ODER-Verknüpfung → MAX-Operator

Anwendung auf regelbasierte Systeme

Systeme werden bei Anwendung des Fuzzy-Konzepts durch Regeln beschrieben. Diese Regeln verknüpfen Eigenschaften der Eingangsgröße und leiten daraus eine Aussage über die Ausgangsgröße ab.

WENN der Winkel im Bereich NL1 UND die Regeldifferenz negativ klein (NK) ist,
DANN soll die Stellgröße null sein (Z).

Die Summe aller Regeln läßt sich am übersichtlichsten in einer Tabelle notieren.

		Eingangsgröße: Bereich des Kurbelwinkels					
		NL1 $\mu = 0,4$	L1 $\mu = 0,6$	L2 $\mu = 0$	NL2 $\mu = 0$	L3 $\mu = 0$	NL3 $\mu = 0$
Ausgangsgröße:	NG $\mu = 0$	NK	NG	NG	NK	NK	NK
	NK $\mu = 0,5$	Z $\mu = 0,4$	NK $\mu = 0,5$	NK	Z	Z	Z
Regelabweichung	Z $\mu = 0,5$	PM $\mu = 0,4$	Z $\mu = 0,5$	Z	PK	PM	PM
	PK $\mu = 0$	PM	PM	PK	PM	PM	PM
$x_w(t) - x_2(t)$	PM $\mu = 0$	PG	PM	PM	PG	PM	PG
	PG $\mu = 0$	PGG	PG	PG	PGG	PG	PGG

In den Spalten werden die Fuzzy-Sets der Winkelbereiche und in den Zeilen die der Regelabweichung eingetragen. Aus der Tabelle kann dann eine Aussage über die Stellgröße entnommen werden.

Verbal würden die Regeln wie folgt lauten:

WENN der Winkel im Bereich NL1 **UND** die Regeldifferenz negativ groß (NG) ist,
DANN soll die Stellgröße negativ klein sein (NK).

ODER

WENN der Winkel im Bereich NL1 **UND** die Regeldifferenz negativ klein (NK) ist,
DANN soll die Stellgröße null sein (Z).

ODER

usw.

In dieser verbalen Aufstellung werden die Operatoren UND und ODER benutzt. An dieser Stelle werden die MIN- oder MAX-Operatoren angewendet.

Beispiel:

Regelabweichung: $\mu = 0,5$ Zugehörigkeit zu NK und $\mu = 0,5$ Zugehörigkeit zu Z
Kurbelwinkel: $\mu = 0,4$ Zugehörigkeit zu NL1 und $\mu = 0,6$ Zugehörigkeit zu L1

Diese Zuordnungen wurden bereits in der Tabelle eingetragen.

WENN Regelabweichung 0,5 NK UND Kurbelwinkel 0,4 NL1, ...

Es wird das Minimum gebildet. Es folgt also

..., DANN Stellgröße 0,4 Z.

In der Tabelle tritt für die Stellgröße einmal 0,4 Z und einmal 0,5 Z auf. Die Regeln sind untereinander durch ODER verknüpft. Es wird also das Maximum gebildet.

Stellgröße $\mu = 0,5$ Z (null)
 $\mu = 0,5$ NK (negativ klein)
 $\mu = 0,4$ PM (positiv mittel)

Defuzzifizierung

Nach der Inferenz erhält man die Zugehörigkeit der Ausgangsgröße zu den einzelnen Fuzzy-Sets. Aus dieser Information muß wieder ein exakter Wert gebildet werden. Dazu werden die Flächen betrachtet, die von den Zugehörigkeitsfunktionen der Fuzzy-Sets bis zum entsprechenden Wert für μ eingeschlossen werden.

Es existieren verschiedene Methoden, um von dieser Fläche auf den entgeltigen Crisp-Wert zu schließen. An dieser Stelle werden nur zwei wichtige Verfahren beschrieben.

MOM (MEAN OF MAXIMA) - Methode

Bei diesem Verfahren werden nur die Regeln mit höchstem Erfüllungsgrad betrachtet. Aus dem arithmetischem Mittel der Crisp-Werte der aktiven Regeln bestimmt man dann die Ausgangsgröße.

$$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i^{\max} \quad (1)$$

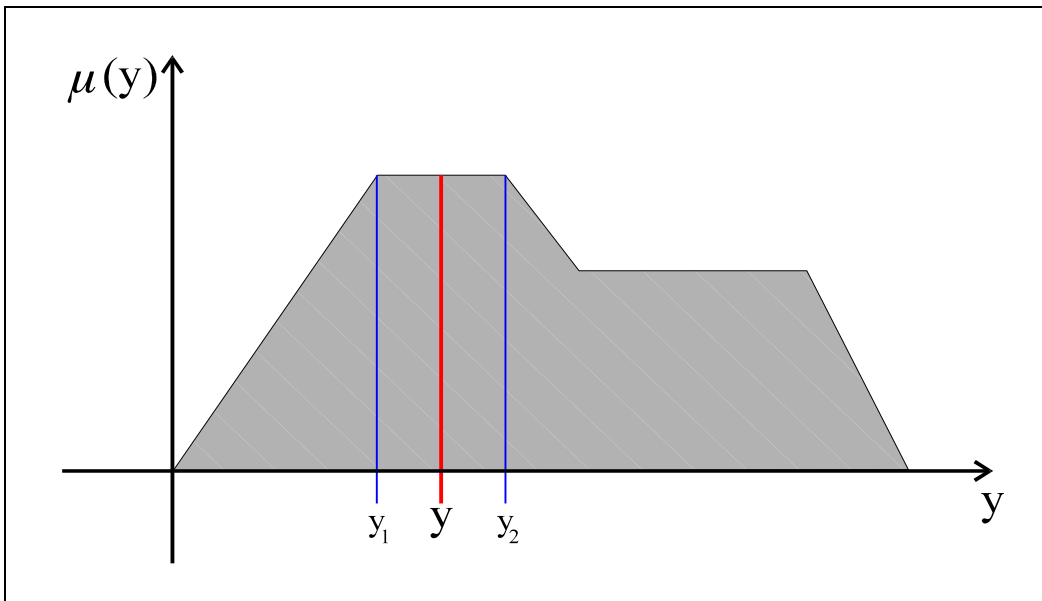


Abbildung Anhang B -5 [5]: Defuzzifizierung nach der MOM - Methode

COA (Centre of area) - Methode

Bei dieser Methode wird die Ausgangsgröße aus dem Abzissenwert des Flächenschwerpunktes berechnet.

$$\bar{y} = \frac{\int_a^b y \cdot \mu_{\text{art}}(y) dy}{\int_a^b \mu_{\text{art}}(y) dy} \quad (2)$$

Diese Methode ist numerisch aufwendig, wird in der Praxis aber am häufigsten benutzt.

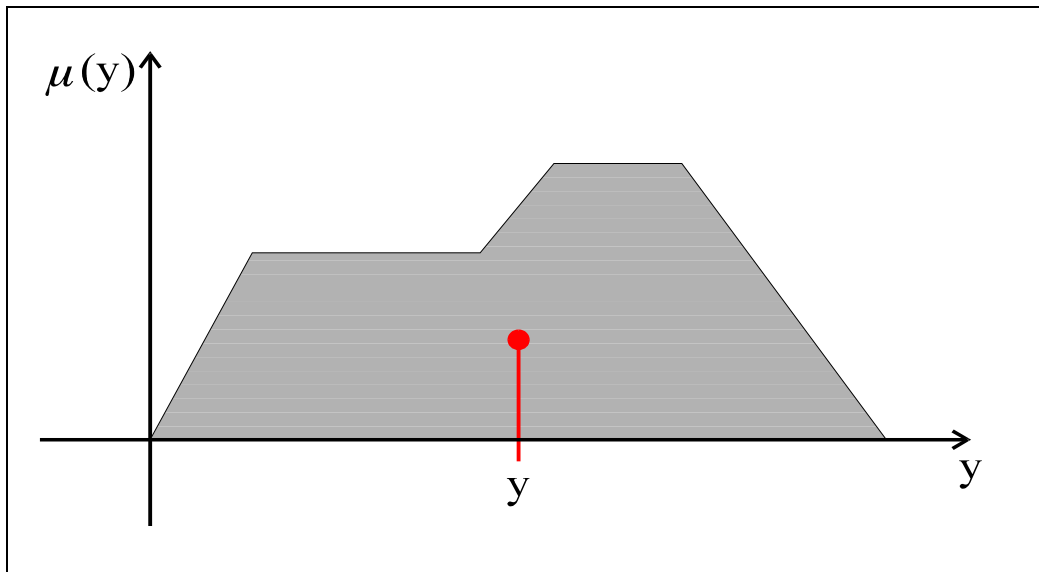


Abbildung Anhang B -6 [5]: Defuzzifizierung nach der COA - Methode

Rechnerisch einfacher wird die COA-Methode, wenn man die sich überschneidenden Schnittflächen doppelt berücksichtigt. Im diesem Fall entfällt die aufwendige Bestimmung der Schnittpunkte der sich überlagernden Zugehörigkeitsfunktionen. Dieses Verfahren wird in dieser Arbeit zur Berechnung der Schaltfläche angewendet.

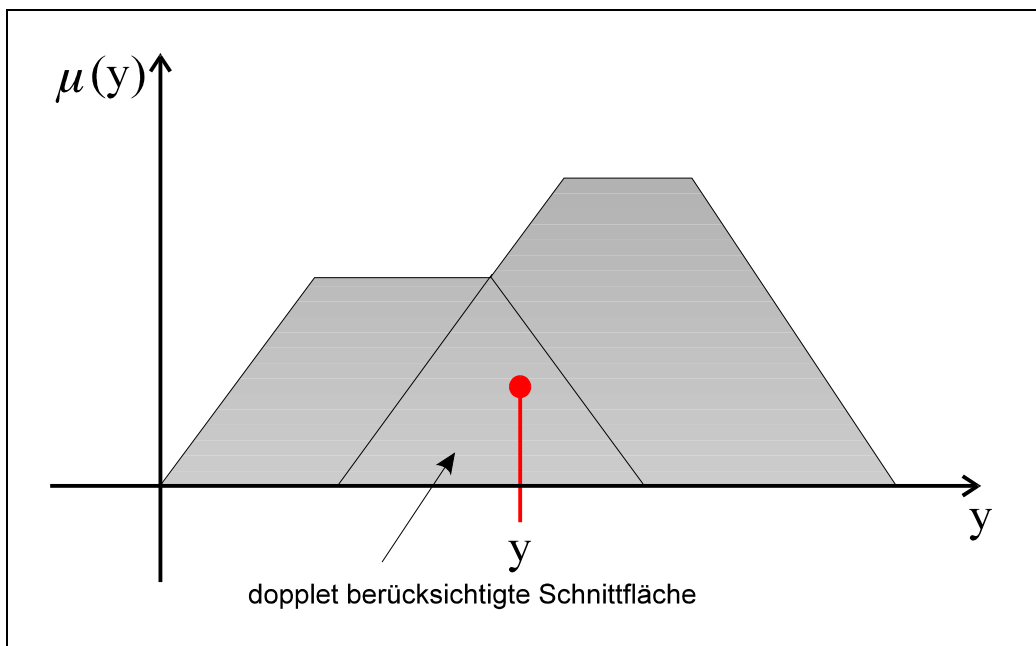


Abbildung Anhang B -7 [5]: COA - Methode mit doppelt berücksichtigter Schnittfläche

Beispiel:

Für die Stellgröße ergab sich die folgende Zuordnung:

- $\mu = 0,5$ Z (null)
- $\mu = 0,5$ NK (negativ klein)
- $\mu = 0,4$ PM (positiv mittel)

Die resultierenden Flächen wurden in Abbildung Anhang B -8 eingetragen. Der Flächenschwerpunkt liegt über dem Abzissenwert 3 V.

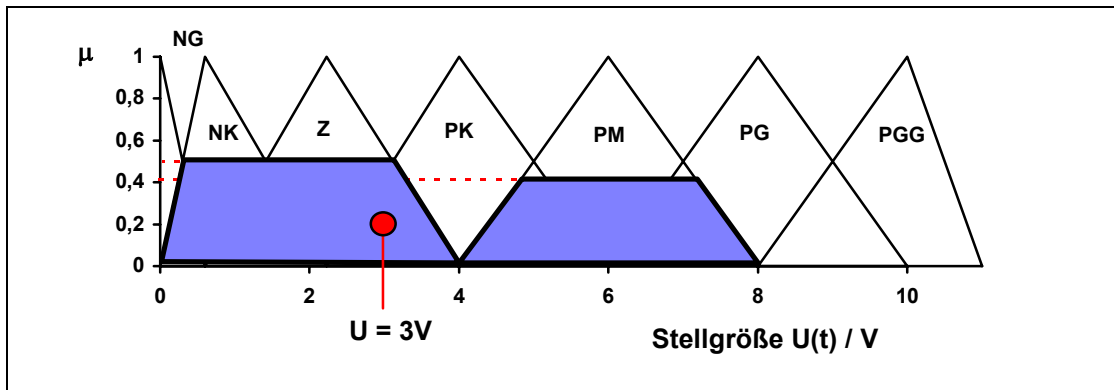


Abbildung Anhang B -8: Fuzzy-Sets für die Stellgröße

Anhang C : S-Functions

S-Functions sind Unterprogramme, die in einem mit SIMULINK erstellten Modell verwendet werden können. Sie werden von dem Funktionsblock „S-Function“ aus der SIMULINK-Bibliothek aufgerufen.

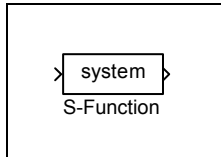


Abbildung Anhang C -1: Block S-Function

Dabei ist zu beachten, daß S-Functions nach festen Konventionen programmiert werden müssen. Die Zustände, Ein- und Ausgangsgrößen müssen definiert werden. Auf bestimmte Systemanfragen hin müssen definierte Parameter übergeben werden. Dazu werden vom übergeordneten Programm Flags übergeben:

flag = 0	Die Initialisierungsparameter für die Zustände müssen übergeben werden.
flag = 2	Die neuen Zustände müssen übergeben werden.
flag = 3	Die Ausgangsgrößen müssen übergeben werden.

Dabei werden die folgenden Vektoren verwendet:

sys[]	Vektor zur Übergabe der Parameter
u[]	Vektor enthält die Eingangsvariablen
x[]	Vektor der Zustände

Für detaillierte Informationen sei auf das Handbuch zu SIMULINK [11] verwiesen.

Meßwertaufbereitung

In dieser S-Function werden die Werkstückposition und die Schlittenposition aufbereitet. Außerdem wird am Ende eines jeden Positioniervorgangs ein Löshsignal für den Integrator des PID-Reglers erzeugt.

Für die Werkstückposition gilt:

1. Erreicht der Schlitten den Schnittpunkt x_S , wird die aktuelle, absolute Position x_{diff} zwischengespeichert.
2. Bis zum Winkel $\Phi_S + 22,5^\circ$ muß noch Gleichlauf mit dem Werkstück herrschen. Der Bezugspunkt wird noch beibehalten.
3. Nach dem Erreichen dieses Winkels wird von der absoluten Position die oben zwischengespeicherte abgezogen und die Differenz $x_S - l_W$ addiert.

$$x_{W,rel}(t) = x_{W,abs}(t) - x_{diff} + x_S - l_W$$

Die Schlittenposition wird wie folgt errechnet:

- Berechnung der Schlittenposition aus dem Winkel:

$$x(t) = 0,6\text{m} - 0,012\text{m} \cdot \sin^2 \Phi - 0,12\text{m} \cdot \cos \Phi$$

- Umrechnung in die Position $x_2(t)$:

$$x_2(t) = \begin{cases} 2 x_0 - 2 x_u + x(t) & \text{für } \Phi_s + 22,5^\circ \leq \Phi(t) < 180^\circ \\ 2 x_0 - x(t) & \text{für } 180^\circ \leq \Phi(t) < 360^\circ \\ x(t) & \text{für } 0^\circ \leq \Phi(t) < \Phi_s + 22,5^\circ \end{cases}$$

Listing:

```
%
function [sys, x0] = xconvert(t,x,u,flag,lw)

%
% Konstante
%
xo = 0.48;      %Umkehrpunkte des Schlittens
xu = 0.72;
xs = 0.6;      %Schnittpunkt

L = 0.6;      %Länge der Pleuelstange
R = 0.12;    %Kurbelradius

fis1=1.671;   % Schnittwinkel
fis2=2.064;   % Schnittwinkel + 22,5°

%
% Folgezustände berechnen
%
if abs(flag) == 2,
%
% Zustände
%
x(1)  Position, von der der Schlitten noch bis zum Schnittpunkt
      absolut verfahren muß
xges = x(1);

%
x(2)  Werkstückposition relativ
xwrel = x(2);

%
x(3)  Flag für Sägeschnitt
flg = x(3);

%
x(4)  Letzte Schnittposition
lastxs = x(4);

%
x(5)  Neue Schnittposition
newxs = x(5);

%
% Eingangsgrößen
%
u(1)  Position des Werkstücks
xw = u (1);

%
u(2)  Winkelstellung der Kurbel
fi = u (2);
```

```
%  
  
% Position x(t) des Schlittens  
xt = L - R^2/2/L*sin(fi)^2 - R*cos(fi);  
  
% Schlitten befindet sich direkt vor dem Schnittpunkt  
if (fi>=0) * (fi<fis1),  
    xges = xt;  
    flg = 0;  
  
% Schlitten befindet sich direkt hinter dem Schnittpunkt  
elseif (fi>=fis1) * (fi<fis2),  
    xges = xt;  
    if (flg==0),  
        flg=1;  
        newxs = xw-(xu+xo)/2+lw;  
    end  
  
% Schlitten befindet sich direkt hinter dem Schnittpunkt+22,5°  
elseif (fi>=fis2) * (fi<pi),  
    xges = 2*xo-2*xu+xt;  
    lastxs = newxs;  
  
% Schlitten ist auf dem Rückweg  
else  
    xges = 2*xo-xt;  
end  
  
% relative Position des Werkstücks  
xwrel = xw-lastxs;  
  
%Zustände übergeben  
sys(1)= xges;  
sys(2)= xwrel;  
sys(3)= flg;  
sys(4)= lastxs;  
sys(5)= newxs;  
  
% Ausgabe der Zustände  
elseif flag == 3,  
    sys(1)= x(2);  
    sys(2)= x(1);  
  
% Löshsignal für Integrator erzeugen  
  
fi = u(2);  
if (fi>=fis2) * (fi<fis2+0.01),  
    sys(3)=1;  
else  
    sys(3)=0;  
end
```

```

elseif flag == 0,
    % Initialisierungen vornehmen
    %
    % Schlittenposition : 2*x0 - 2*xu + 0,6475
    %
    % Werkstückposition : 0,6475 - L
    %
    % Flag für Schnitt : 1 (neue Schnittpos. wurde gesetzt)
    %
    % Letzte Schnittpos.: 0
    %
    % Neue Schnittpos. : 0

    x0 = [0.1675 (0.6475-lw) 1 0 0];

    % 5 kontinuierliche Zustände
    % 0 diskrete Zustände
    % 3 Ausgänge
    % 2 Eingänge
    sys = [0 5 3 2 0 1];

else
    sys = [];
end

```

Drehzahlvorsteuerung

Diese S-Function berechnet den Offsetwert der Drehzahlvorsteuerung.

$$u_{\text{OFFSET}}(t) = \frac{v(t)}{K_V \cdot R \cdot \left[\sin(\Phi(t)) - \frac{E}{L} \cos(\Phi(t)) - \frac{R}{2 \cdot L} \sin(2\Phi(t)) \right]}$$

Dieser Wert wird dann auf u_{MAX} begrenzt.

Listing:

```

%
function [sys, x0] = offset(t,x,u,flag,max)

%
% Konstante
%
L = 0.6;           %Länge der Pleuelstange
R = 0.12;          %Kurbelradius
K = 5.513;         %Kreisverstärkung

if flag== 3,
% Ausgabe

    v = u(1);      % Geschwindigkeit Werkstück
    fi = u(2);     % Winkelder Kurbel

% fi auf das Intervall [0, pi] begrenzen
    if (fi>pi),
        fi = 2*pi-fi;
    end

% Winkel, die zu einer Stellgröße >10V führen würden vorab behandeln
    if (fi<0.3),
        sys=max;
    end
end

```

```
elseif (fi>=2.95),
    sys=max;
% Offsetgröße berechnen
else
    omega =v/( R* (sin(fi)- R/2/L*sin(2*fi)) );
    sys = omega/K;

    if (sys>max),          % auf Maximalwert begrenzen
        sys = max;
    end
end

elseif flag == 0,
    % Initialisierungen vornehmen
    x0 = [];

    % 0 kontinuierliche Zustände
    % 0 diskrete Zustände
    % 1 Ausgang
    % 2 Eingänge
    sys = [0 0 1 2 0 1];

else
    sys = [];
end
```

Sollwertgenerator

Mit dieser S-Function wird aus der aktuellen Werkstückposition $x_w(t)$ ein Sollwert $x_g(t)$ für den P-Regler generiert.

$$x_g(t) = \begin{cases} x_w(t) & \text{für } 0^\circ \leq \Phi(t) < \Phi_s + 22,5^\circ \\ x_w(t) + x_f & \text{für } \Phi_s + 22,5^\circ \leq \Phi(t) < 360^\circ \end{cases}$$

Listing:

```
function [sys, x0] = generat(t,x,u,flag,off)

%
%   Konstante
%
%   fis1=1.671;           % Schnittwinkel
%   fis2=2.064;         % Schnittwinkel + 22,5°
%
%   Ausgabe der Zustände
%
if flag == 3,

    fi = u(1);
    xw = u(2);

%   Schlitten befindet sich direkt hinter dem Schnittpunkt+22,5°
    if (fi>=fis2) * (fi<2*pi),
        sys = xw+off;

%   Schlitten befindet sich vor dem Schnittpunkt
    else
        sys = xw;
    end

elseif flag == 0,
%   Initialisierungen vornehmen
    x0 = [];

%   1 kontinuierliche Zustände
%   0 diskrete Zustände
%   1 Ausgänge
%   2 Eingänge
    sys = [0 0 1 2 0 1];

else
    sys = [];
end
```

Anhang D : Animationsprogramm

Zum Überprüfen der gefundenen mathematischen Zusammenhänge und zur Veranschaulichung des Betriebs eines Pleuelgetriebes wurde ein kurzes C-Programm entwickelt. Mit diesem Programm kann eine Animation der Pleuelbewegung durchgeführt werden. Dabei können Parameter verändert werden.

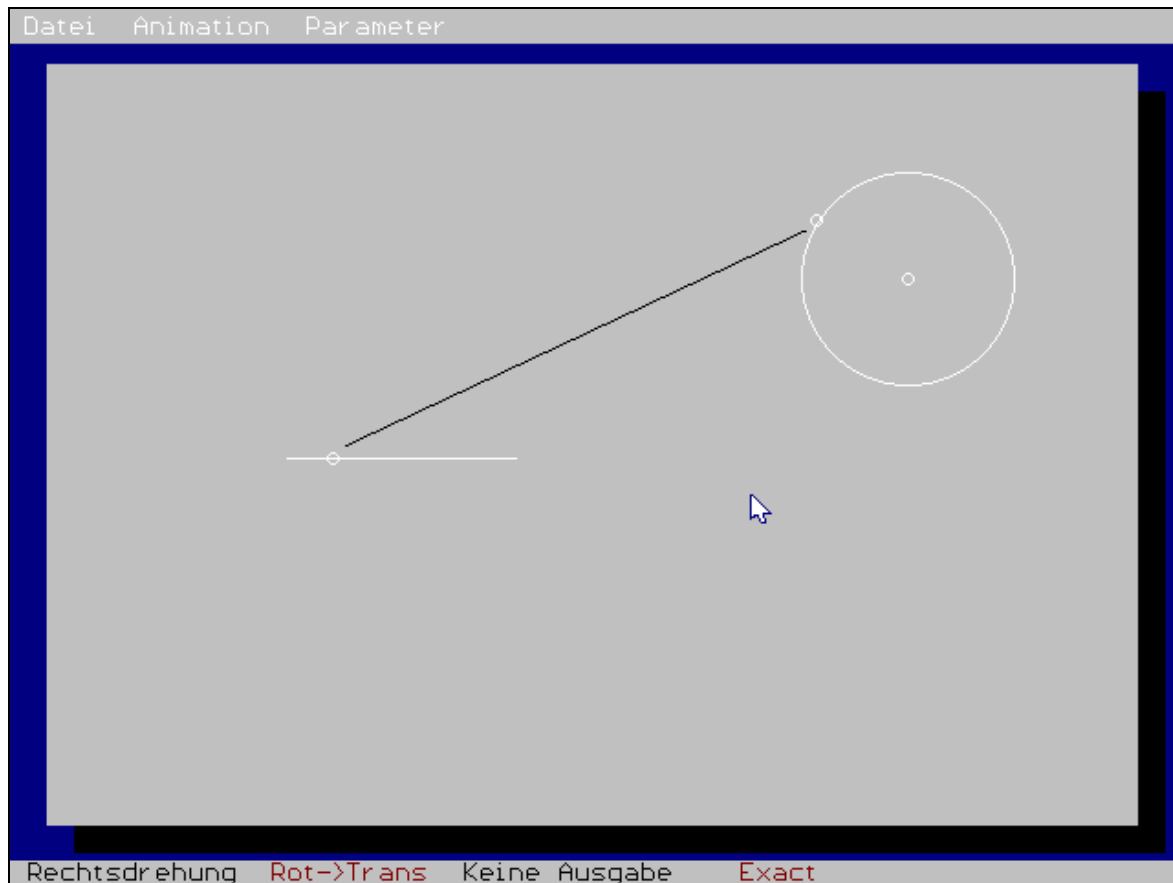


Abbildung Anhang D -1: Bildschirmaufbau des Animationsprogramms

Über Pulldownmenus können die folgenden Funktionen ausgeführt werden:

Datei-Menu

SPEICHERN und LADEN wurde nicht implementiert.

NEUSTART springt zurück zum Startbildschirm.

ENDE verläßt das Programm.

Animation-Menu

ANIMATION beendet und startet die Bewegung des Pleuelgetriebes.

Parameter-Menu

RICHTUNG legt fest, welche Transformationsgleichungen verwendet werden sollen.

Trans->Rot bewegt den Sägeschlitten gleichförmig und errechnet die korrespondierende Kurbelstellung.

Rot->Trans dreht die Kurbel gleichförmig und errechnet die korrespondierende Schlittenposition.

PARAMETER ermöglicht die Einstellung von vier Parametern.

Exzentrizität, Radius Kurbel und **Länge Pleuelstab** sind geometrische Parameter des Pleuelgetriebes.

Mit **Schrittweite** kann die Genauigkeit eingestellt werden, mit der sich das Getriebe bewegt. Je kleiner die Schrittweite umso langsamer ist auch die Animationsbewegung.

DREHSINN legt fest, ob sich die Kurbel **rechts** oder **links** herum dreht.

AUSGABE bestimmt, ob der Kurbelwinkel und die korrespondierende Schlittenposition in eine Datei gespeichert werden sollen. Dies ermöglicht z.B. eine Ausgabe der Daten mit einem Plotprogramm.

NÄHERUNG dient der Einstellung, ob zur Berechnung Näherungsformeln verwendet werden sollen oder die exakten Gleichungen. So war es möglich die Genauigkeit der Näherungsgleichungen zu überprüfen.

In der Statuszeile am unteren Bildschirmrand werden die jeweils aktuellen Einstellungen angezeigt.