Proceedings of the 2007 IEEE
International Conference on Mechatronics and Automation
August 5 - 8, 2007, Harbin, China

# Design and realization of fuzzy self-tuning PID speed controller based on TMS320F2812 DSPs

Yanpeng Dou
*Automation department of Inner Mongolia University*
*Hohhot 010021*
douyanpeng@sina.com

Zhang Ze
*Automation department of Inner Mongolia University*
*Hohhot 010021*
auto@imu.edu.cn
zhangzeimu@163.com

***Abstract*** **- This paper discussed how we should apply fuzzy self-tuning PID controller to AC-speed adjustable system, and emphasized how to realize this controller using fuzzy search table method in C++ language based on TMS320F2812, which is the newest 32-bit fixed point DSP from TI company. The author eventually applied this controller to PMSM-speed adjustable vector control system in MCK2812, which is a motion control development kit of technosoft company. And the results show that this controller can improve dynamic characteristics and static characteristics of system obviously.**

***Index Terms - Fuzzy self-tuning PID controller, TMS320F2812, AC-speed adjustment***

## I. Introduction

In recent years, people apply fuzzy self-tuning PID controller to speed control of AC motor gradually. It is very effective for realizing higher accuracy, higher speed as well as better dynamic characteristics and static characteristics in AC-speed adjustable system. But AC-speed adjustment algorithm needs to implement complex coordinate transformation for realizing uncoupling of system model, and fuzzy self-tuning PID controller needs to on-line calculate corrections of three PID parameters, so people must use high performance processor to ensure real-time running of program. So far, TMS320F2812, which is the newest 32-bit fixed point DSP from TI company, is the best DSP chip in the domain of motion control. Its improved harvard structure suits the high speed calculation and signal processing extremely, and on-chip two event managers can simplify the peripheral electric circuit, so its use provides the guarantee for realization of fuzzy self-tuning PID control algorithm in AC-speed adjustable system.

## II. Design of the Controller

### *A Basic structure of fuzzy self-tuning PID control*

AC-speed adjustable system pays attention to control performance of rotational speed of motor, so we take rotational speed error and rotational speed error change rate as input variables of fuzzy self-tuning PID controller. The corrections of three PID parameters $\Delta K_p, \Delta K_i$ and $\Delta K_d$ are output variables of controller.

Fuzzy self-tuning PID control algorithm on-line adjusts three PID parameters in order to satisfy different error and error change rate to PID parameter's different request through finding the fuzzy relation among three PID parameters, speed error and rotational speed error change rate as well as fuzzy logic control principle.

In Fig.1, r (t) is reference input; e is speed error; $\Delta e$ is speed error change rate; E and $\Delta E$ are fuzzy sets of reflecting Language variables of system error and error change rate; U (t) is final output of fuzzy PID controller.

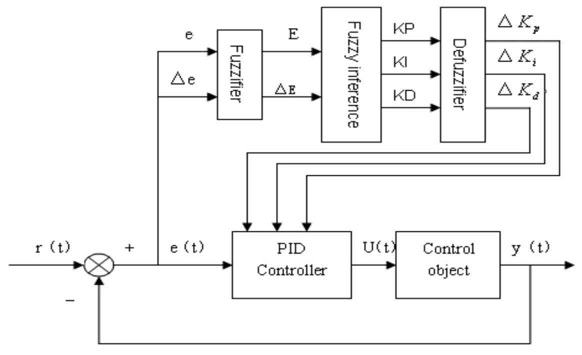


Fig.1 Schematic diagram of fuzzy self-tuning PID

Principle of $K_p, K_i$ and $K_d$ self-tuning is: according to error and error change rate of controlled object in sampling time, we can confirm the size of the parameter correction. Proper attention to both static characteristics of controlled object (higher or lower than given value) and dynamic characteristics of response (close or deviate given value). Its algorithm is that converting input variables into fuzzy variables, then matching them with fuzzy rule in knowledge base. If they matches some rules, executing the results of these rules.

Discrete form of conventional PID controller is as follows:

$$u(k) = K_p e(k) + \frac{K_p T_s}{T_i}\sum_{i=0}^{k-1} e(i) + K_p T_d \frac{e(k)-e(k-1)}{T_s} \qquad (1)$$

Where $T_s$ is sampling period, $T_i$ is integral time constant, $T_d$ is differential time constant, and the following relations are tenable:

$$K_i = \frac{K_p T_s}{T_i}; \qquad K_d = \frac{K_p T_d}{T_s}; \qquad (2)$$

Where $K_p, K_i$ and $K_d$ can be denoted by the following formula:

$$K_p = K_{p0} + \Delta K_p$$
$$K_i = K_{i0} + \Delta K_i$$
$$K_d = K_{d0} + \Delta K_d \tag{3}$$

Where $K_{p0}$ , $K_{i0}$ and $K_{d0}$ are conventional PID parameters, $\Delta K_p, \Delta K_i$ and $\Delta K_d$ are the crisp corrections.

*B Confirmation of fuzzy language variable value, membership functions and fuzzy control rules.*

*1. Confirmation of language variable value*

When selecting language variable value, proper attention to both flexibility of control rules and request of easy and feasible. This paper take PMSM-speed adjustable system as an example, two input language variables E and $\Delta$E as well as three output language variables $\Delta K_p$ , $\Delta K_i$ and $\Delta K_d$ are defined as {NL NS ZE PS PL}, which stand for {negative large, negative small, zero, positive small, positive large}. According to actual situation, the integral domain of $E$ , $\Delta E$ , $\Delta K_p$ , $\Delta K_i$ and $\Delta K_d$ are defined as [-6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6]. So language variable values are divided into five fuzzy sets as follows:

- Negative large (NL)—located nearby -6
- Negative small(NM)—located nearby -2
- Zero(ZE)—located nearby 0
- Positive small(PS)—located nearby 2
- Positive large(PL)—located nearby 6

In addition, we need to convert continuous domain into integral domain, the transformation can complete through the following formula:

$$\begin{cases} b = q(a - \dfrac{X_L + X_H}{2}) \\ q = \dfrac{2n}{X_H - X_L} \end{cases} \tag{4}$$

Where $a$ is one of the continuous domain between $X_L$ and $X_H$ , $b$ is one of the integral domain corresponds to $a$ , $q$ is scaling factor used for fuzzification.

*2. Confirmation of the membership functions*

| | -6 | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NL | 1 | 0.8 | 0.7 | 0.4 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| NS | 0.1 | 0.3 | 0.5 | 0.7 | 1 | 0.8 | 0.3 | 0 | 0 | 0 | 0 | 0 | 0 |
| ZE | 0 | 0 | 0 | 0 | 0 | 0.5 | 1 | 0.5 | 0 | 0 | 0 | 0 | 0 |
| PS | 0 | 0 | 0 | 0 | 0 | 0 | 0.3 | 0.8 | 1 | 0.7 | 0.5 | 0.3 | 0.1 |
| PL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.1 | 0.4 | 0.7 | 0.8 | 1 |

Table.1 Membership functions evaluation table

Because shape of membership function is steeper the resolution is higher, for better robustness of system, we use triangular membership function whose resolution is higher when error approaches to zero, and we use Gaussian membership function whose resolution is lower when error is bigger.

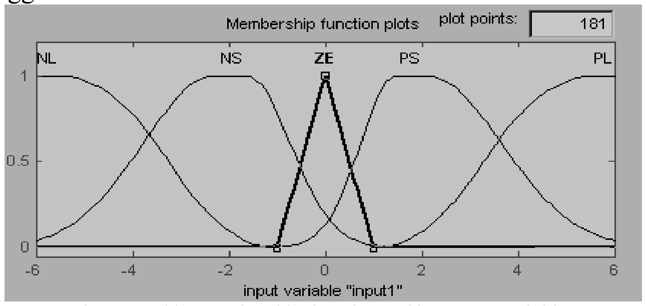


Fig.2 Graphic Membership functions of language variables

*3. Fuzzy rules and fuzzy inference*

In the process of system control, considering inference time and simplification of operational method, we implement the inference using Max-Min method of Mamdani. If $\mu_E$ , $\mu_{\Delta E}$ , $\mu_{\Delta Kp}$ , $\mu_{\Delta Ki}$ and $\mu_{\Delta Kd}$ stand for membership functions of $E$ , $\Delta E$ , $\Delta K_p$ , $\Delta K_i$ and $\Delta K_d$ , the action intensity of the ith rule is:

$$\alpha_i = \min \{\mu_{Ei}(\text{E}),\ \mu_{\Delta Ei}(\Delta \text{E})\}$$

Through "the operator of taking smaller" of Mamdani, we can obtain decision-making of the ith rule as follows:

$$\mu_{\Delta Kpi}(s) = \text{Min}(\alpha_i, \mu_{\Delta Kpi}(s))$$
$$\mu_{\Delta Kii}(s) = \text{Min}(\alpha_i, \mu_{\Delta Kii}(s))$$
$$\mu_{\Delta Kdi}(s) = \text{Min}(\alpha_i, \mu_{\Delta Kdi}(s))$$

According to above the formulas, membership functions of output status S are obtained as follows:

$$\mu_{\Delta Kp}(s) = \text{Max}(\mu_{\Delta Kpi}(s))$$
$$\mu_{\Delta Ki}(s) = \text{Max}(\mu_{\Delta Kii}(s))$$
$$\mu_{\Delta Kd}(s) = \text{Max}(\mu_{\Delta Kdi}(s))$$

| Kp\Ki\Kd ΔE / E | NL | NB | ZE | PS | PL |
|---|---|---|---|---|---|
| NL | PL\NL\PS | PL\NL\ZE | PL\NL\ZE | PS\NL\ZE | ZE\ZE\PS |
| NB | PL\NL\NL | PS\NS\NL | PS\NS\NS | ZE\ZE\ZE | NL\PS\PS |
| ZE | PS\NS\NL | PS\NS\NS | ZE\ZE\NS | NS\PS\ZE | NL\PS\PS |
| PS | PS\NS\NL | ZE\ZE\NS | NS\PS\NS | NS\PS\ZE | NL\PL\PS |
| PL | ZE\ZE\PS | NS\PS\ZE | NS\PS\ZE | NL\PL\ZE | NL\PL\PS |

Table.2 Fuzzy rules of $\Delta K_p, \Delta K_i$ and $\Delta K_d$

The defuzzification method is the centroid method, we may obtain integer output as follows:

$$y = \frac{\sum_{k=1}^{m} v_k \mu_v(v_k)}{\sum_{k=1}^{m} \mu_v(v_k)} \tag{5}$$

Where m is output quantificational index, and the actual output can be obtained by the following formulas:

$$\begin{cases} a = k[b + \dfrac{n(X_L + X_H)}{X_H - X_L}] \\ k = \dfrac{X_H - X_L}{2n} \end{cases} \tag{6}$$

Where $b$ is one of the integral domain between $-n$ and $n$, $a$ is one of the continuous domain between $X_L$ and $X_H$, $k$ is scaling factor used for defuzzification.

## III. Realization of Fuzzy Self-tuning PID Speed Controller

### A. *Hardware project*

TMS320F2812, which was promoted by TI company in 2003, is a high performance 32-bit fixed point DSP. So far, this chip's performance is the best in the domain of motion control. Comparing this chip with C240x series chip, TMS320F2812 has some unique Characteristics as follows:

- High-Performance 32-Bit CPU, High-Performance Static CMOS Technology—150 MHz (6.67-ns Cycle Time).
- 1.8V core voltage, 3.3V I/O voltage, the loss is very low.
- On-chip memory contains up to 128K×16 flash and up to 18K x 16 SARAM as well as 4K ROM. A 12-bit ADC.
- Development tools include ANSI C/C++ Compiler/ Assembler/Linker, Code Composer Studio IDE, DSP/BIOS, JTAG scan controller. Programmer may develop application in high-level program language such as C language and C++ language, it will reduce the cycle of software design greatly and ensure realization of the highest compiling efficiency.
- Event manager core is compatible with C240x series, so users of C24x series and 240x series may use TMS320F2812 conveniently.

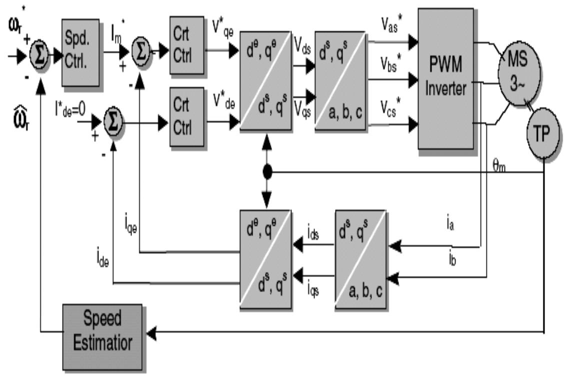


Fig.3 PMSM vector control schematic diagram

Taking PMSM vector control speed adjustable system as an example, system hardware is divided into the following two parts: control panel and driver panel. Control panel consists of TMS320F2812 chip, memory, voltage transformation chip and peripheral connector. Driver panel consists of IPM and related parts. Encoder, which is used for speed sensor, obtains actual speed through on-chip QEP circuit. Hall sensor is used for detecting the current of main loop, and sampling signals are sent in on-chip ADC after clamp and filter, which are used for current feedback signals after A/D conversion.

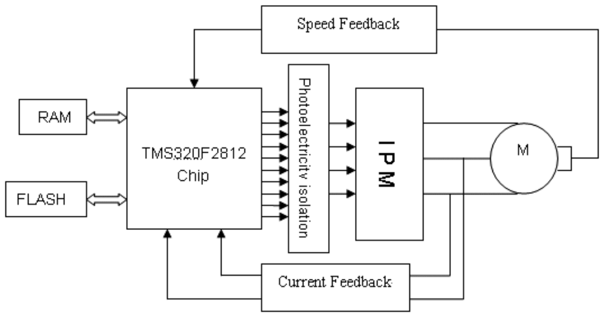


Fig.4 Hardware structure diagram

### B. *Software project*

TMS320F2812 supports C/C++ language, so we program in C language.

$\Delta K_p$, $\Delta K_i$ and $\Delta K_d$ is obtained through searching fuzzy search table in order to simplify program structure and reduce calculational load. The fuzzy search table is obtained by the formula from the second chapter.

*I. Confirmation of the offset*

Completing search of fuzzy search table through program language, the key is how to confirm the position of output in fuzzy search table. Fuzzy search table may be regarded as two-dimensional $m \times n$ matrix, where $m = 13$, $n = 13$. TMS320F2812 on-chip data memory is arranged according to one-dimensional space order, so we need to convert two-dimensional matrix constituted by fuzzy search table into one-dimensional array, and deposit it in the memory in order. Then, we use "Initial address + offset" method to confirm the position of output in the memory. Offset can be confirmed by the following formula:

$$\text{Offset} = 13 \times (E + 6) + (\Delta E + 6) \tag{7}$$

Where $E$ is error which is located in the integral domain. $\Delta E$ is error change rate which is located in integral domain. For example, if $E$=-4 and $\Delta E$=5, offset=13 × (-4+6) + (5+6) = 37. The address of output is "Initial address + 37" in the memory.

*II. Realizing fuzzy PID controller in C language*

Realizing search of fuzzy search table through C language. We may define three arrays such as a[169], b[169], c[169], which stand for search table of $\Delta K_p$, search table of $\Delta K_i$, search table of $\Delta K_d$ respectively. Finally, obtaining output through "output address = initial address of array + offset".

In the process of using program language to realize fuzzy self-tuning PID controller, we must pay attention to the following several questions:

- Limiting amplitude of fuzzy rules, the input language variable will be limited between -6 and 6.
- Limiting amplitude of output of PID controller.
- After speed adjustment, current speed error variable, the previous speed error variable and earlier speed error variable, whose values must be exchanged each other in order to make preparations for next PID adjustment.

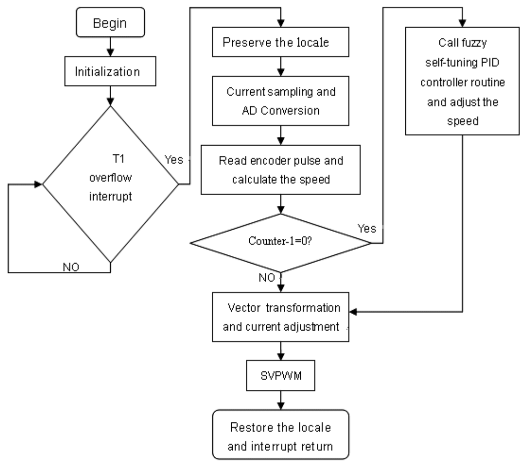
| Kp\Ki\Kd Edelta E | -6 | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -6 | 6\-6\2 | 6\-6\-2 | 6\-6\-2 | 5\-5\-4 | 4\-4\-6 | 4\-4\-6 | 4\-4\-5 | 2\-2\-5 | 2\-2\-5 | 1\-1\-4 | 0\0\-3 | 0\0\-2 | 0\0\-1 |
| -5 | 6\-6\2 | 6\-6\-2 | 6\-6\-2 | 5\-5\-4 | 4\-4\-6 | 4\-4\-6 | 3\-3\-4 | 2\-2\-4 | 2\-2\-4 | 1\-1\-3 | 0\0\-2 | -1\0\-1 | -1\0\0 |
| -4 | 6\-6\1 | 6\-6\-1 | 6\-6\-2 | 5\-5\-4 | 4\-4\-5 | 4\-4\-5 | 2\-2\-4 | 2\-2\-3 | 2\-2\-3 | 1\-1\-3 | 0\0\-2 | -1\0\-1 | -2\0\0 |
| -3 | 5\-6\0 | 5\-6\-1 | 5\-5\-2 | 5\-4\-3 | 4\-3\-4 | 4\-3\-4 | 2\-2\-4 | 1\-1\-2 | 1\-1\-2 | 0\0\-2 | -1\1\-2 | -2\1\-1 | -2\1\0 |
| -2 | 4\-6\0 | 4\-5\-1 | 4\-4\-2 | 4\-3\-3 | 4\-2\-4 | 4\-2\-4 | 2\-2\-4 | 0\0\-2 | 0\0\-2 | -1\1\-2 | -2\2\-2 | -2\2\-1 | -2\2\0 |
| -1 | 4\-6\0 | 4\-5\-1 | 4\-4\-2 | 4\-3\-2 | 3\-2\-2 | 4\-2\-2 | 2\-2\-2 | 0\0\-2 | 0\0\-2 | -1\1\-2 | -2\2\-2 | -2\2\-1 | -3\2\0 |
| 0 | 4\-4\0 | 4\-4\0 | 3\-4\0 | 3\-3\0 | 2\-2\0 | 2\-2\0 | 0\0\0 | -2\2\0 | -2\2\0 | -3\3\0 | -4\4\0 | -4\4\0 | -4\4\0 |
| 1 | 2\-4\0 | 2\-3\0 | 1\-2\0 | 1\-1\0 | 0\0\0 | 0\0\0 | -2\2\0 | -2\2\0 | -2\2\0 | -3\3\0 | -4\4\0 | -4\5\0 | -4\5\0 |
| 2 | 2\-4\3 | 2\-3\2 | 1\-2\1 | 1\-1\1 | 0\0\1 | 0\0\1 | -2\2\1 | -2\2\1 | -2\2\1 | -3\3\1 | -4\4\1 | -4\5\2 | -4\6\3 |
| 3 | 2\-2\6 | 2\-2\4 | 0\-1\2 | 0\0\2 | -1\1\2 | -1\1\2 | -3\2\2 | -3\3\2 | -3\3\2 | -4\4\2 | -4\5\2 | -5\6\4 | -5\6\6 |
| 4 | 2\0\6 | 1\0\5 | -1\0\3 | -1\1\3 | -2\2\3 | -2\2\3 | -4\2\3 | -4\4\2 | -4\4\2 | -4\5\2 | -4\6\2 | -5\6\4 | -6\6\6 |
| 5 | 1\0\6 | 1\0\5 | -2\0\4 | -2\1\4 | -3\2\4 | -3\2\4 | -4\3\4 | -4\4\2 | -4\4\2 | -5\5\2 | -5\6\2 | -6\6\4 | -6\6\6 |
| 6 | 0\0\2 | 0\0\-2 | -2\0\-2 | -2\1\-4 | -4\2\-6 | -4\2\-6 | -4\4\-6 | -4\4\-6 | -4\4\-6 | -5\5\-5 | -6\6\-4 | -6\6\-3 | -6\6\-2 |

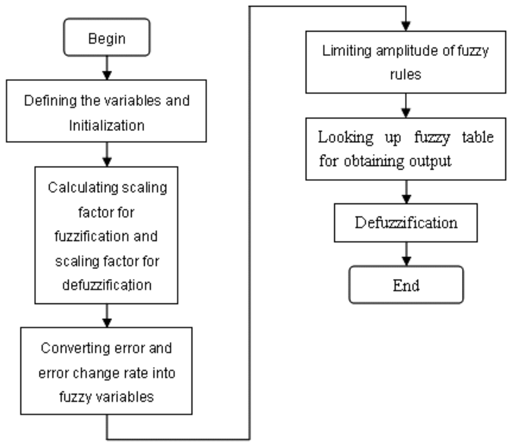
Table.3 Fuzzy search table

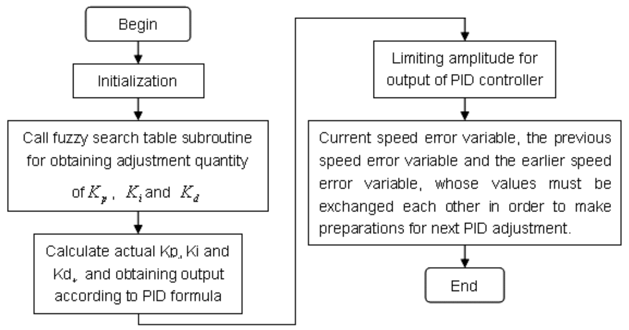
CCS2.2 is used for program development environment of TMS320F2812. Before debugging program, we need add head files, C source files and command file to project. In this program, fuzzy search table is located in DRAMH0 block whose address area is 0x3f9000h-0x3fA000h. Because three arrays are defined as local variable, the Initial addresses of three arrays are not changeless. As a result of length limit, I only list program flow diagram and core source code as follows:

Fuzzy search table source code:

```
q1=12/(Xh_e-Xl_e);
q2=12/(Xh_delta_e-Xl_delta_e);
k1=(Xh_Kp-Xl_Kp)/12;
k2=(Xh_Ki-Xl_Ki)/12;
k3=(Xh_Kd-Xl_Kd)/12;
PID_e0=PID_reference-N;
Fuzzy_PID_e=PID_e0*q1;
PID_delta_e=PID_e0-PID_e1;
Fuzzy_PID_delta_e=PID_delta_e*q2;
if(Fuzzy_PID_e>6)
Fuzzy_PID_e=6;
else if(Fuzzy_PID_e<-6)
Fuzzy_PID_e=-6;
if(Fuzzy_PID_delta_e>6)
Fuzzy_PID_delta_e=6;
else if(Fuzzy_PID_delta_e<-6)
Fuzzy_PID_delta_e=-6;
a=13*(Fuzzy_PID_e+6)+(Fuzzy_PID_delta_e+6)+&Kp[0];
b=13*(Fuzzy_PID_e+6)+(Fuzzy_PID_delta_e+6)+&Ki[0];
c=13*(Fuzzy_PID_e+6)+(Fuzzy_PID_delta_e+6)+&Kd[0];
Fuzzy_Kp=*a;
Fuzzy_Ki=*b;
Fuzzy_Kd=*c;
Kp_fuzzy_out=k1*Fuzzy_Kp;
Ki_fuzzy_out=k2*Fuzzy_Ki;
Kd_fuzzy_out=k3*Fuzzy_Kd;
```


Begin
Initialization
T1 overflow interrupt
Yes
NO
Preserve the locale
Current sampling and AD Conversion
Read encoder pulse and calculate the speed
Counter-1=0?
Yes
NO
Call fuzzy self-tuning PID controller routine and adjust the speed
Vector transformation and current adjustment
SVPWM
Restore the locale and interrupt return
Fig.5 Main program flow diagram


Begin
Defining the variables and Initialization
Calculating scaling factor for fuzzification and scaling factor for defuzzification
Converting error and error change rate into fuzzy variables
Limiting amplitude of fuzzy rules
Looking up fuzzy table for obtaining output
Defuzzification
End
Fig.6 Fuzzy search program flow diagram


Begin
Initialization
Call fuzzy search table subroutine for obtaining adjustment quantity of $K_p$, $K_i$ and $K_d$
Calculate actual Kp, Ki and Kd, and obtaining output according to PID formula
Limiting amplitude for output of PID controller
Current speed error variable, the previous speed error variable and the earlier speed error variable, whose values must be exchanged each other in order to make preparations for next PID adjustment.
End
Fig.7 PID speed adjustment program flow diagram

PID speed adjustment source code:

```
PID_e0=PID_reference-N;
A_coeff=(Kp_fuzzy_out+Kp_init)+(Ki_fuzzy_out+Ki_init)+(
Kd_fuzzy_out+Kd_init);
B_coeff=2*(Kd_fuzzy_out+Kd_init)+(Kp_fuzzy_out+Kp_init
);
C_coeff=(Kd_fuzzy_out+Kd_init);
U_delta=A_coeff*PID_e0-B_coeff*PID_e1+C_coeff*PID_e2
;
PID_out=PID_out1+U_delta;
if(PID_out>PID_out_MAX)
```

```
PID_out=PID_out_MAX;
else if(PID_out<PID_out_MIN)
PID_out=PID_out_MIN;
PID_e2=PID_e1;
PID_e1=PID_e0;
PID_out1=PID_out;
```

## IV. Simulation

We need to simulate AC-speed adjustable system in Matlab/Simulink environment in order to prove the feasibility of this algorithm. Controlled object of AC-speed adjustable system is a PMSM. Its parameters are as follows:

$L_q = L_d = 0.1295$H;

Flux induced by magnets = 0.6115Wb;

J=$1.56\times10^{-3}$ kg. $m^2$;

P=4;

Vector control speed adjustment algorithm is used in the system. Conventional PID controller and fuzzy self-tuning PID controller are used respectively in the speed loop. The control effect is as follows:

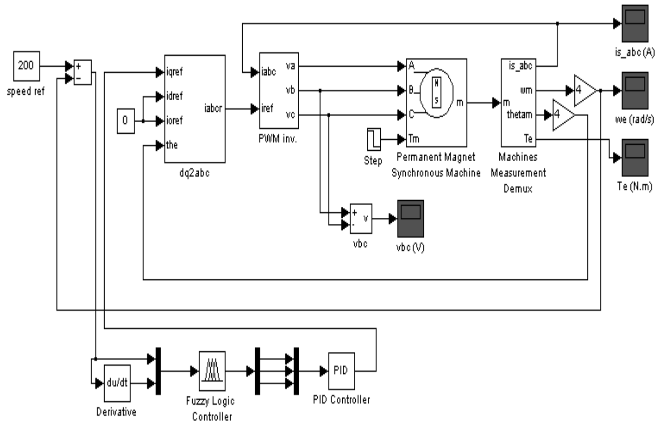


Fig.8 System simulation model

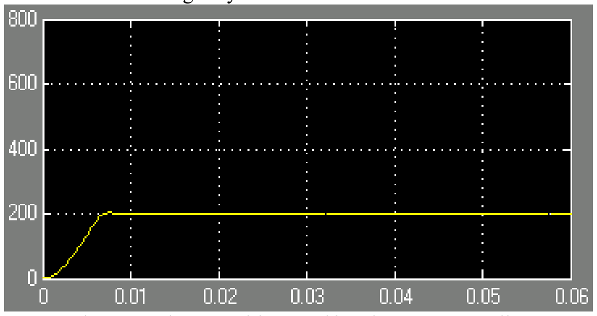


Fig.9 Speed curve of fuzzy self-tuning PID controller

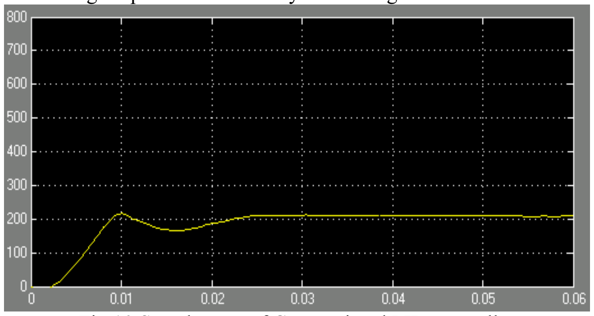


Fig.10 Speed curve of Conventional PID controller

## V. Conclusion

This paper elaborated how to program fuzzy self-tuning PID controller using C++ language based on structure features of TMS320F2812 DSPs and described algorithm principle of fuzzy self-tuning PID controller. As a new special purpose DSPs for motor control, TMS320F2812, which owns perfect structure and high performance, will be mainstream product in the domain of motion control in the future, so it has important significance for realizing fuzzy self-tuning PID control using program language based on TMS320F2812. Finally, the experimental results show that fuzzy self-tuning PID controller can improve dynamic characteristics and static characteristics of system obviously.

## References

[1] Tzafestas S G. Fuzzy systems and fuzzy expert control : An overview [ J ] . *Knowledge Engineering Review* , 1994

[2] TAN Guan-zheng , XIAO Hong-feng , WANG Yue-chao. Optimal fuzzy PID controller with adjustable factors based on flexible polyhedron search algorithm [J]. Journal of Central South University of Technology (in Chinese), 2002, 9(2):128-133.

[3] Dave Misir , Heidar A. Malki , Guanrong Chen. Design and analysis of a fuzzy proportional – integral –derivative controller. Fuzzy Sets and Systems 79 (1996): 297-314.

[4] Gungor Bal , Erdal Bekiroglu , Sevki Demirbas , Ilhami Colak. Fuzzy logic based DSP controlled servo position control for ultrasonic motor. Energy Conversion and Management 45 (2004): 3139–3153.

[5] Liu Fucai, Han Huishan, Chen Li. Program realization of induction motor fuzzy logic control system. Journal of motor and control, 2006, 10 (1):18-22.

[6] H.S. Zhong, T. Shaohua and W.P. Zhuang, Fuzzy self-tuning of PID controller, Fuzzy Sets and Systems 56 (1993) 34-46.