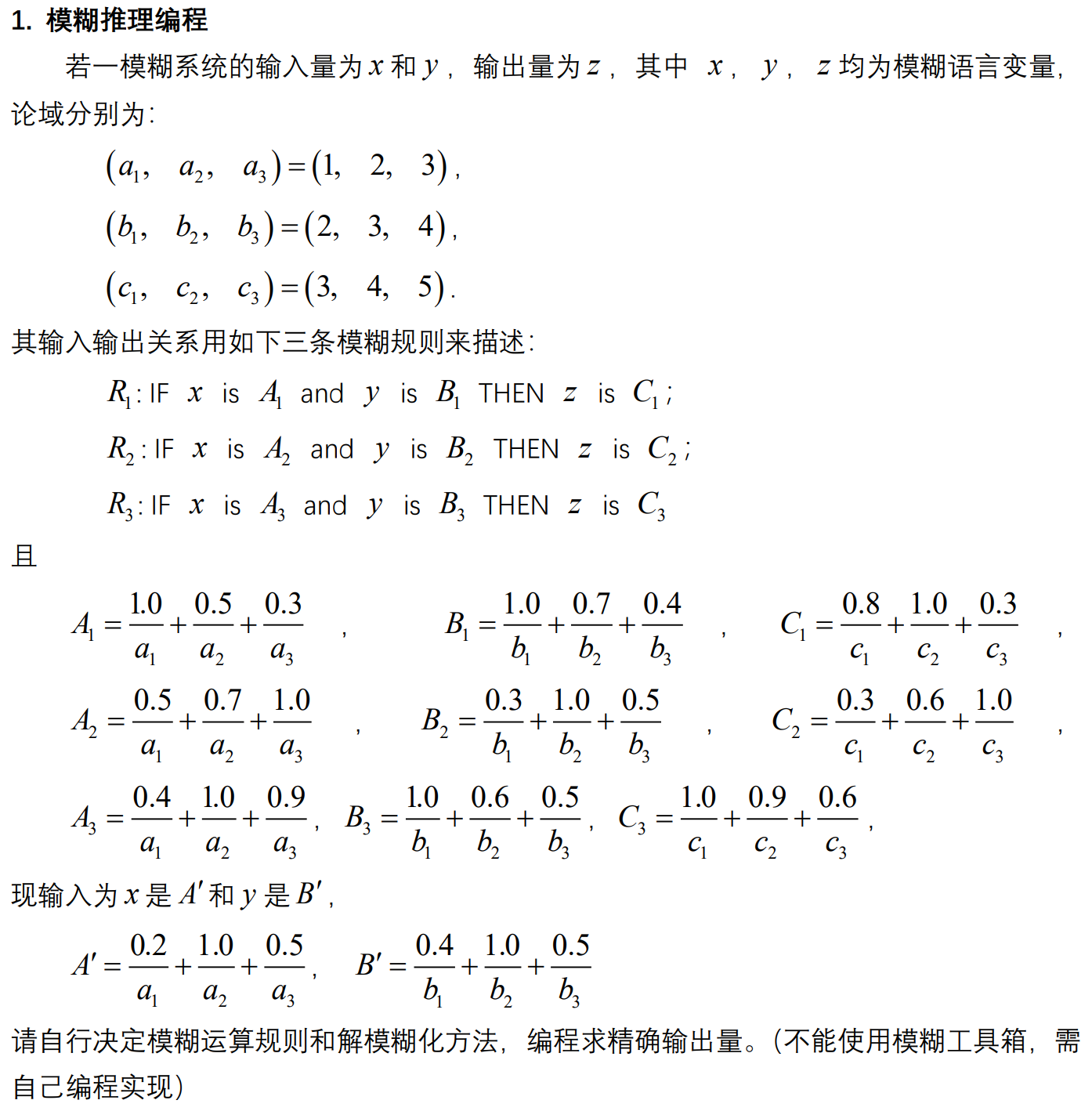
## EX1 模糊推理编程



对于本问题，在关系生成时采用Mamdani方法，输出时采用重心法，当进行以上输入时，可获得输出为：



（代码实现见下页）

% Fuzzy Reasoning & Output

% IC\_ZJU2022FALL\_HW3\_1

% Reasoning (A,B)→C

% created by StvLi LiPeize 2022-12-17

% Reasonin: max-min

% Output : Mamdani

clear; clc;

% Input

A\_ = [0.2 1.0 0.5 ];

B\_ = [0.1 1.0 0.5 ];

% Fuzzy Sets

A = [1.0 0.5 0.3; 0.5 0.7 1.0; 0.4 1.0 0.9];

B = [1.0 0.7 0.4; 0.3 1.0 0.5; 1.0 0.6 0.5];

C = [0.8 1.0 0.3; 0.3 0.6 1.0; 1.0 0.9 0.6];

cdd = [3 4 5];

% similarity between Major & Minor Premise

A\_simi = zeros(1,3);

B\_simi = zeros(1,3);

temp = zeros(1,3);

for i = 1:3 % similarity between A & A\_

temp = min([A\_;A(i,:)]);

A\_simi(1,i) = max(temp);

end

for i = 1:3 % similarity between B & B\_

temp = min([B\_;B(i,:)]);

B\_simi(1,i) = max(temp);

end

% Output

Simi = zeros(3,3); % Similarity Matrix

for i = 1:3

for j = 1:3

Simi(i,j) = min(A\_simi(1,i),B\_simi(1,j));

end

end

nume = 0 ;

deno = 0 ;

for i = 1:3

temp = min([Simi(i,i)\*ones(1,3);C(i,:)]);

nume = nume + sum(cdd.\*temp);

deno = deno + sum(temp);

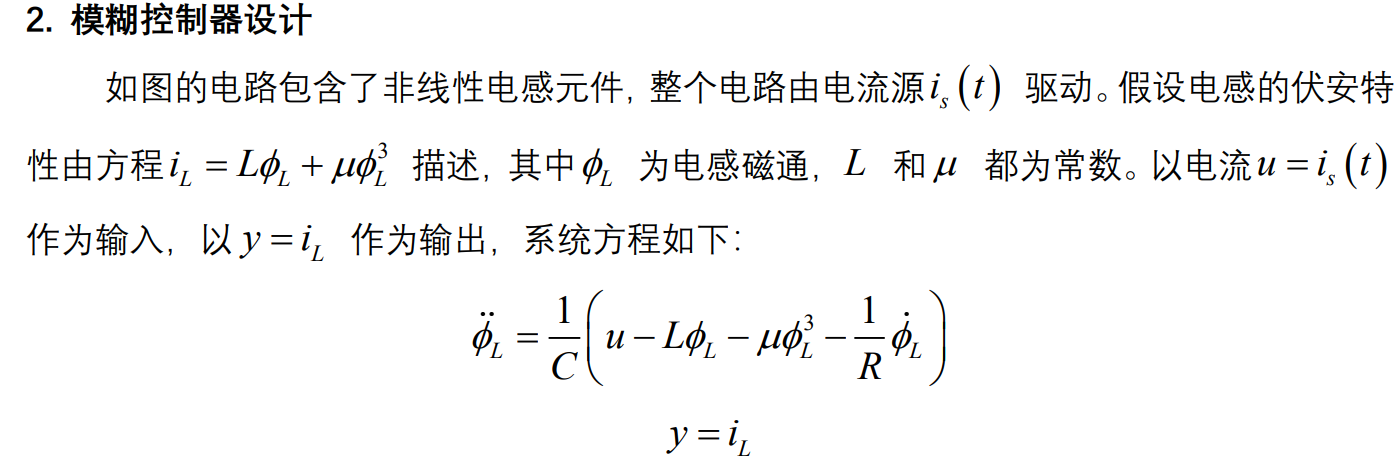
end

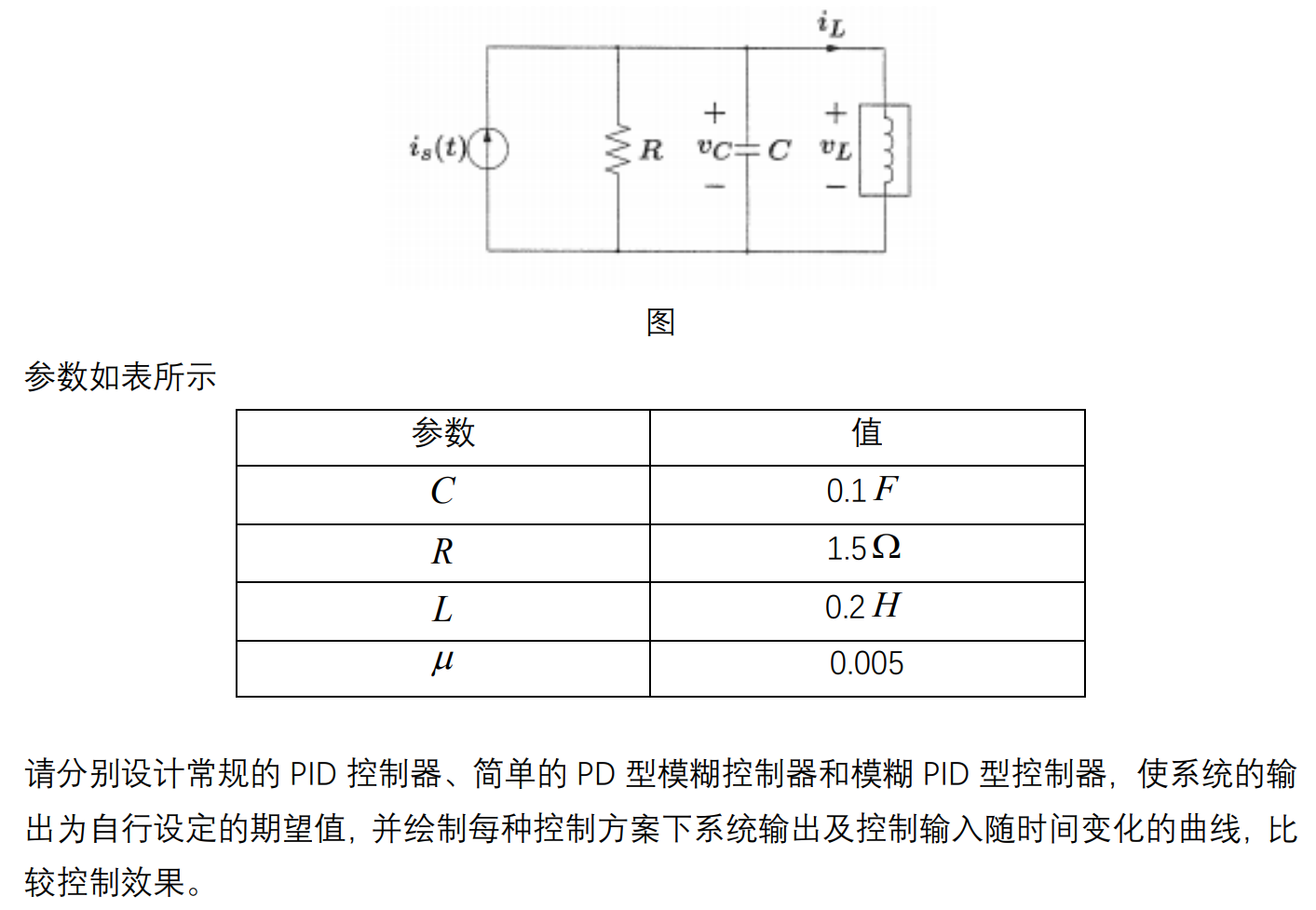
y = nume/deno;

disp("output = ")

disp(y)

## EX2 模糊控制器设计





根据题目中给出的系统方程在s-funt框架内构建被控对象状态方程

%==========================================================================

% mdlDerivatives

% Return the derivatives for the continuous states.

%==========================================================================

%

function sys=mdlDerivatives(t,x,u)

% parameter

C = 0.1;

R = 1.5;

L = 0.2;

miu = 0.005;

% system equation

dx1 = x(2);

dx2 = 1/C\*(u-L\*x(1)-miu\*x(1)^3-1/R\*x(2));

sys = [dx1 dx2]' ;

% end mdlDerivatives

%==========================================================================

% mdlOutputs

% Return the block outputs.

%==========================================================================

%

function sys=mdlOutputs(t,x,u)

% parameter

L = 0.2;

miu = 0.005;

% output equation

i\_L = L\*x(1)+miu\*x(1)^3;

sys = i\_L ;

% end mdlOutputs

系统模型通过simulink进行搭建和调试

### 常规PID控制器设计

PID控制器设计参数：Kp=100 Ki=60 Kd=10 τ=100

使用以上控制器的控制系统的部分瞬态响应性能指标为：

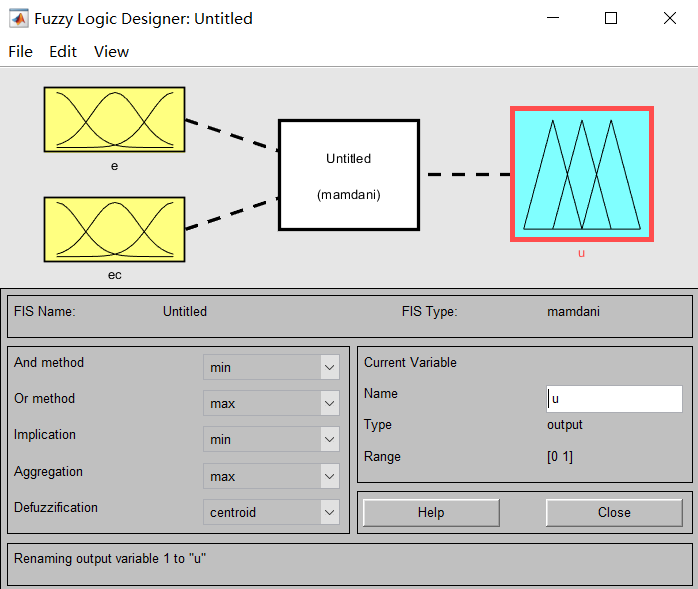
最大超调Mp = 8.6% 稳定时间（△=2%，下同）ts = 0.228s 上升时间tr = 0.06s



以上PID控制器的瞬态性能指标将作为后续模糊控制器性能目标的主要参考

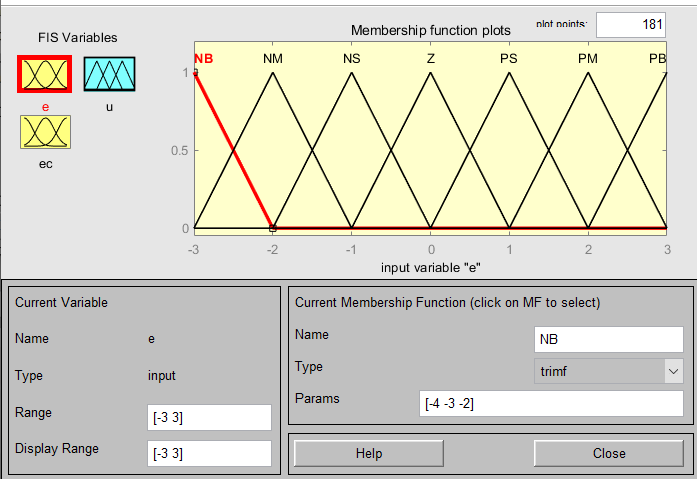
### FuzzyPD控制器

使用matlab中fuzzy control工具箱，输入fuzzy指令打开配置窗口



在窗口中定义And、Or、Implication、Aggregation、Defuzzification的运算规则

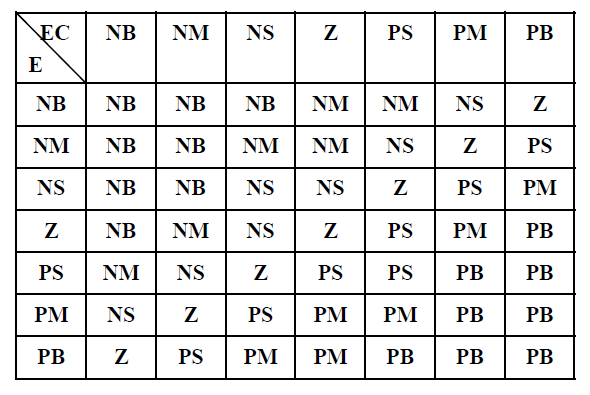
在双击输入隶属度函数和输出隶属度函数的图标分别设置其隶属度函数



其中输入e和ec的范围均为[-3,3]，均分别设置7个不同的模糊集NB NM NS Z PS PM PB，其隶属度函数类型均采取trimf，参数分别设为：[-4 -3 -2],[-3 -2 -1],[-2 -1 0],[-1 0 1],[0 1 2],[1 2 3],[2 3 4]。

输出u的范围亦为[-3,3]，均分别设置7个不同的模糊集NB NM NS Z PS PM PB，其隶属度函数类型均采取trimf，参数分别设为：[-4 -3 -2],[-3 -2 -1],[-2 -1 0],[-1 0 1],[0 1 2],[1 2 3],[2 3 4]。

在edit-Rules中按下表设置模糊规则：

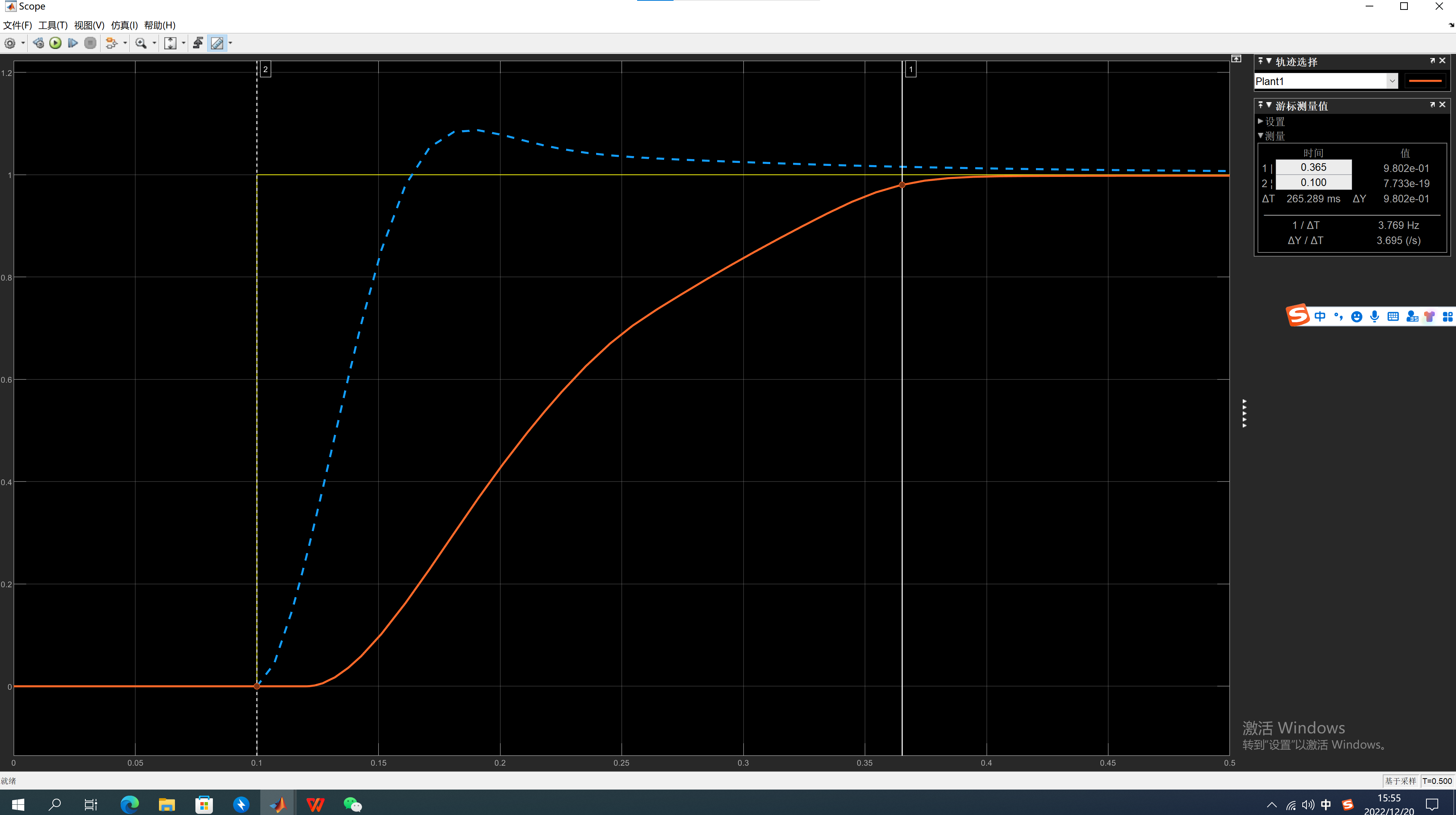


在simulink中搭建并调试，线路图如下：

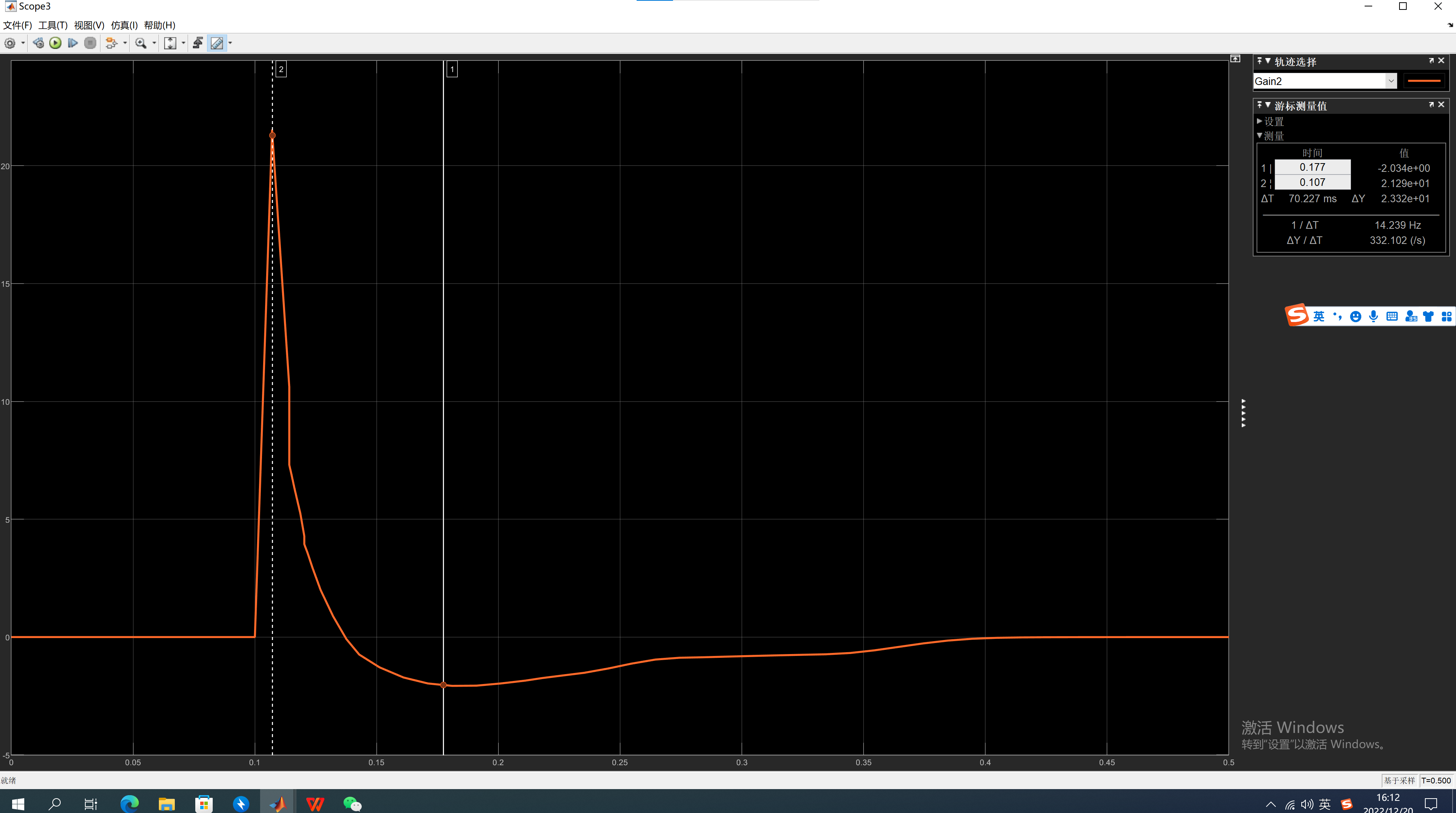
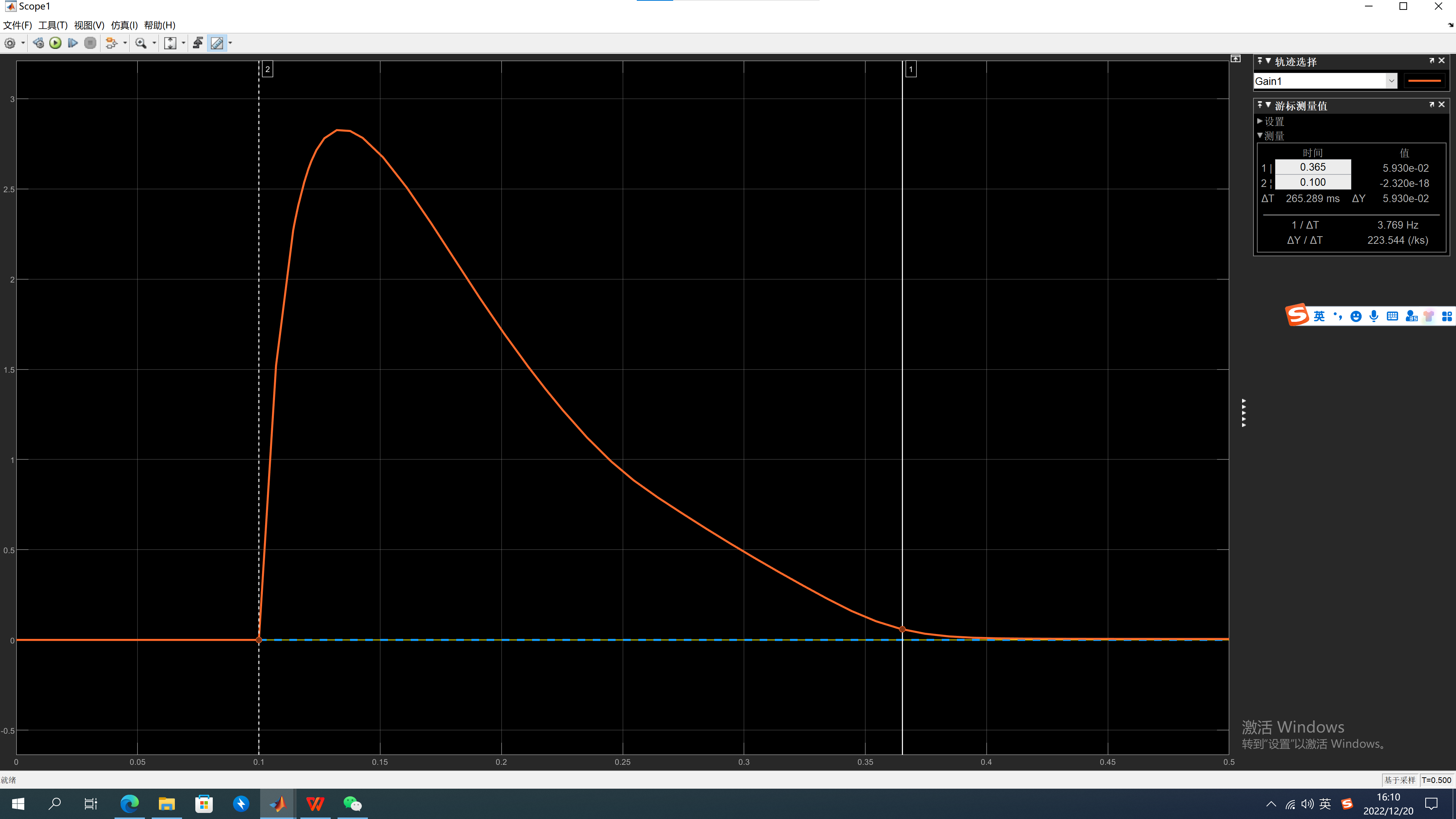


调试过程：

为了保证一阶导输入不发散，在输入模糊控制器前，阶跃信号先经过一个τ=0.01s的一阶低通滤波（下同，不再强调）。在第一轮调试中先以kp=3（这个数据根据阶跃大小为1和输入范围[-3,3]确定，下不做讨论更改，仅展示仿真结果）kd=0.3 Gain，输出增益ku = 50做初步尝试，仿真结果输出如下：



从上图可以看出，这一控制器的稳定时间ts = 0.265s，上升时间tr=0.26s——这一参数较为不理想。观察模糊控制器的输入：

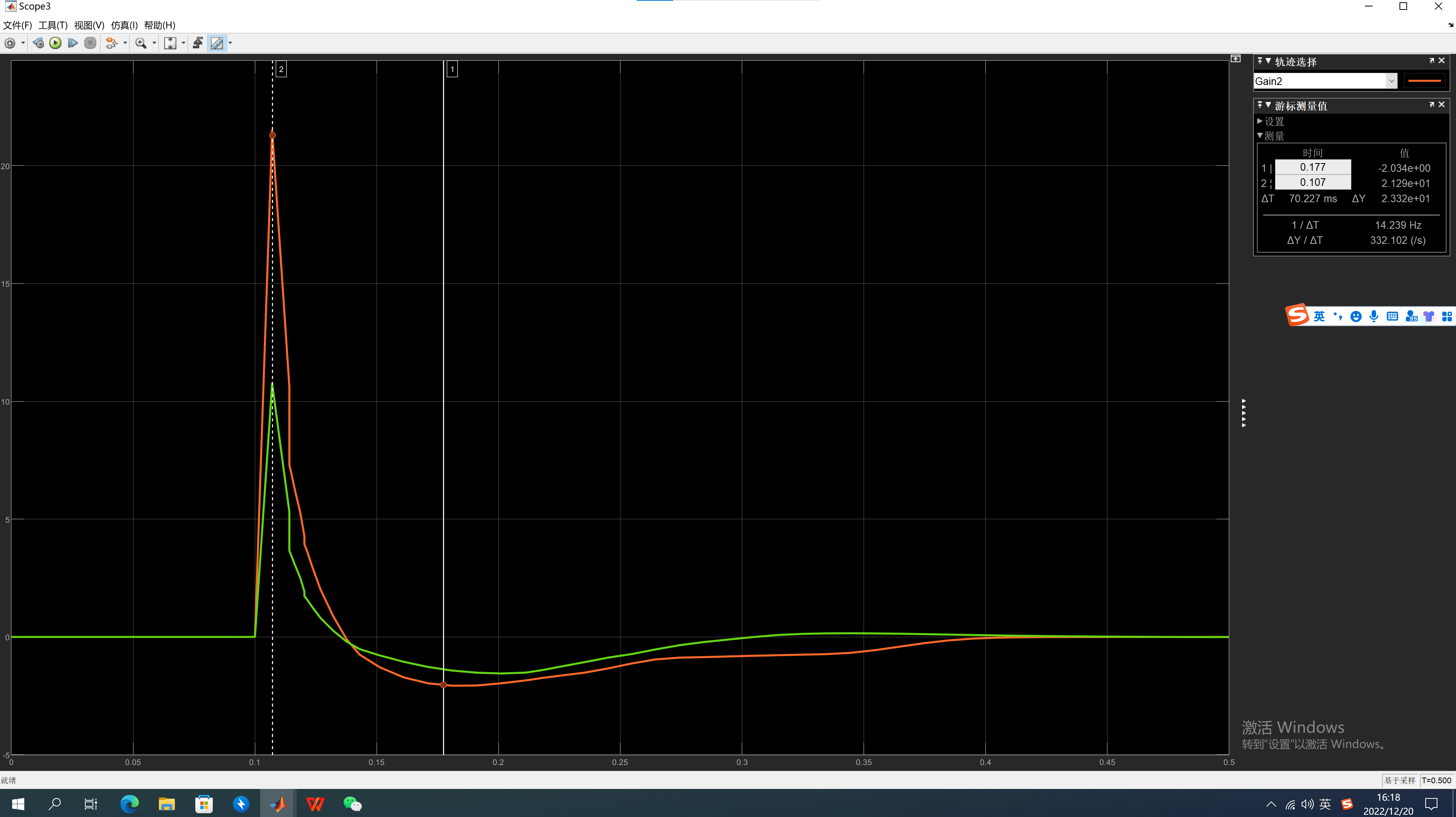
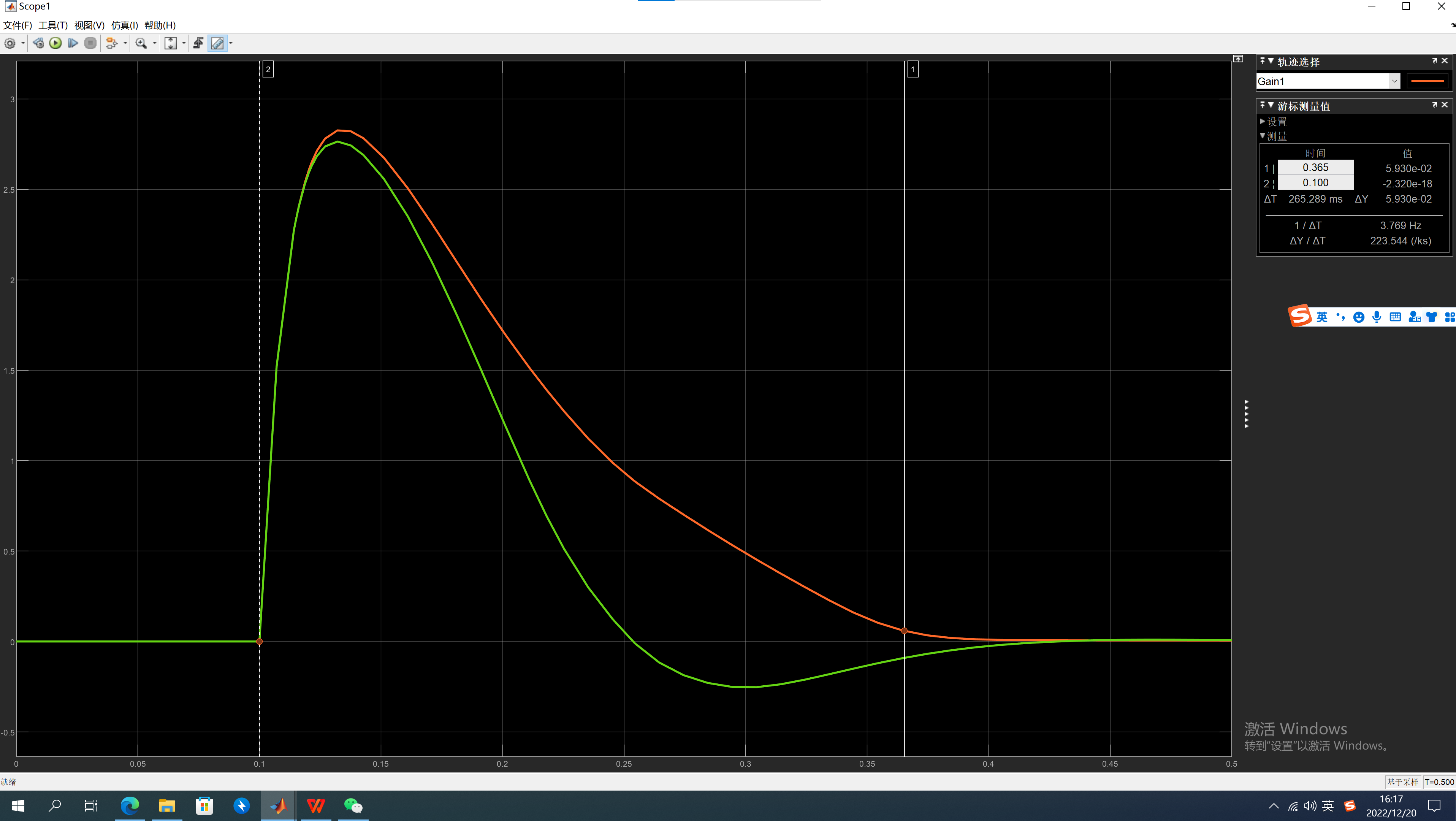


观察到阶跃发生后，模糊控制器一阶导输入快速增加饱和，可能造成模糊控制器没有起到很好地作用效果。修正思路：减小一阶导输入的增益，扩大控制器输出增益。

第二轮调试中，适当减小一阶导的输入增益，设制参数为kp = 3，kd = 0.15，系统输出增益ku=50。系统仿真输出如下图所示，其最大超调Mp=8.4%，稳定时间ts=0.279s，上升时间tr=0.15s，可见其输出上升段动态性能明显改善。



观察模糊控制器的输入参数，可以看出系统输入在初期快速上升的趋势被有效削减了，这使得模糊控制器的模糊规则可以在长时间内按梯次发挥作用，有利于控制系统。

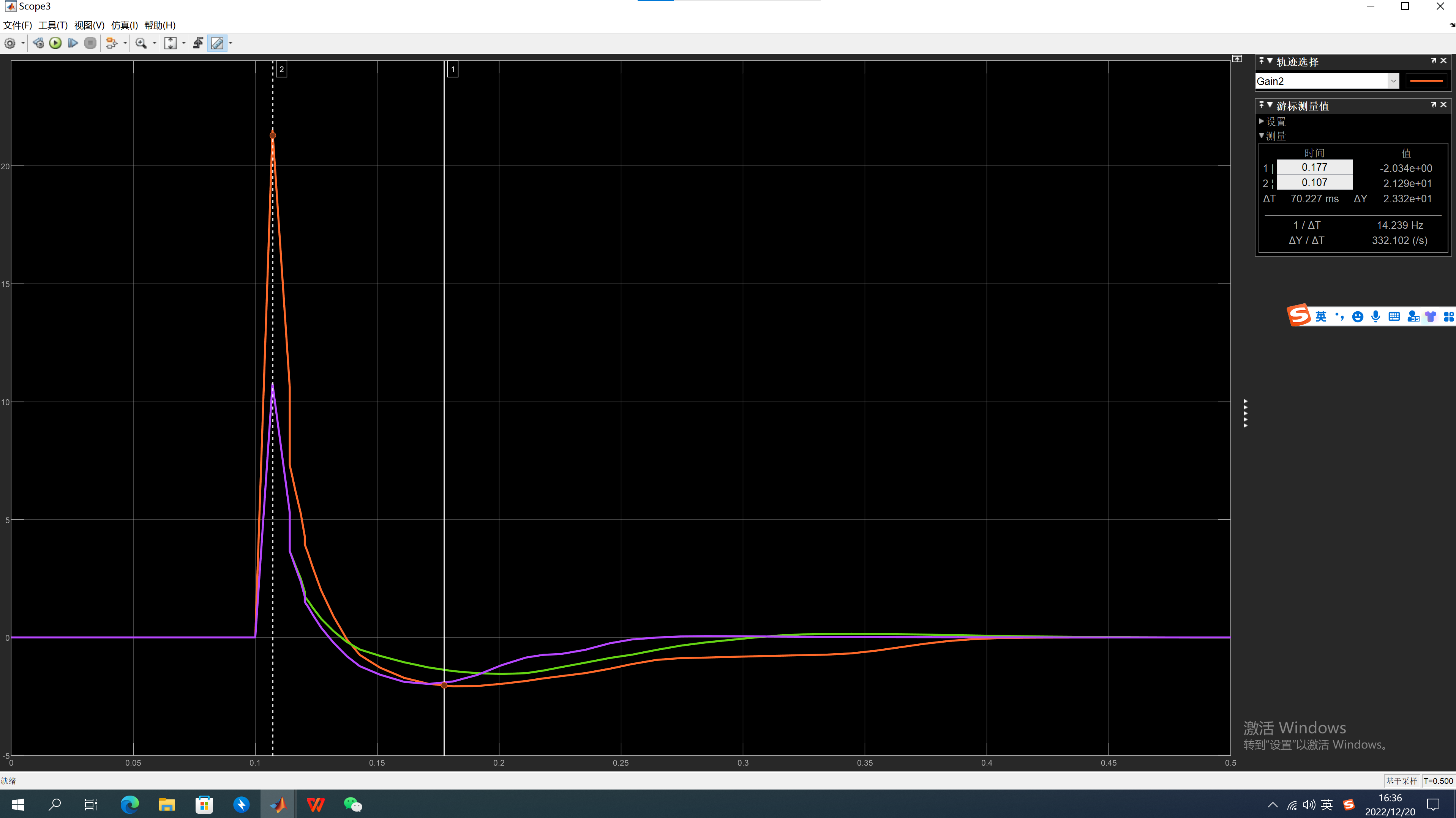


在第三轮调试中，适当增加控制器输出增益，设制参数为kp = 3，kd = 0.15，系统输出增益ku=100。系统仿真输出如下图所示，其最大超调Mp=2.2%，稳定时间ts=0.173s，上升时间tr=0.13s，可见其输出的超调得到了充分抑制、上升段动态性能得到进一步改善，其综合动态性能已明显优于传统PID控制。

为了对比后续FuzzyPID控制，这里需要强调FuzzyPD控制器的稳态误差约为0.2%。

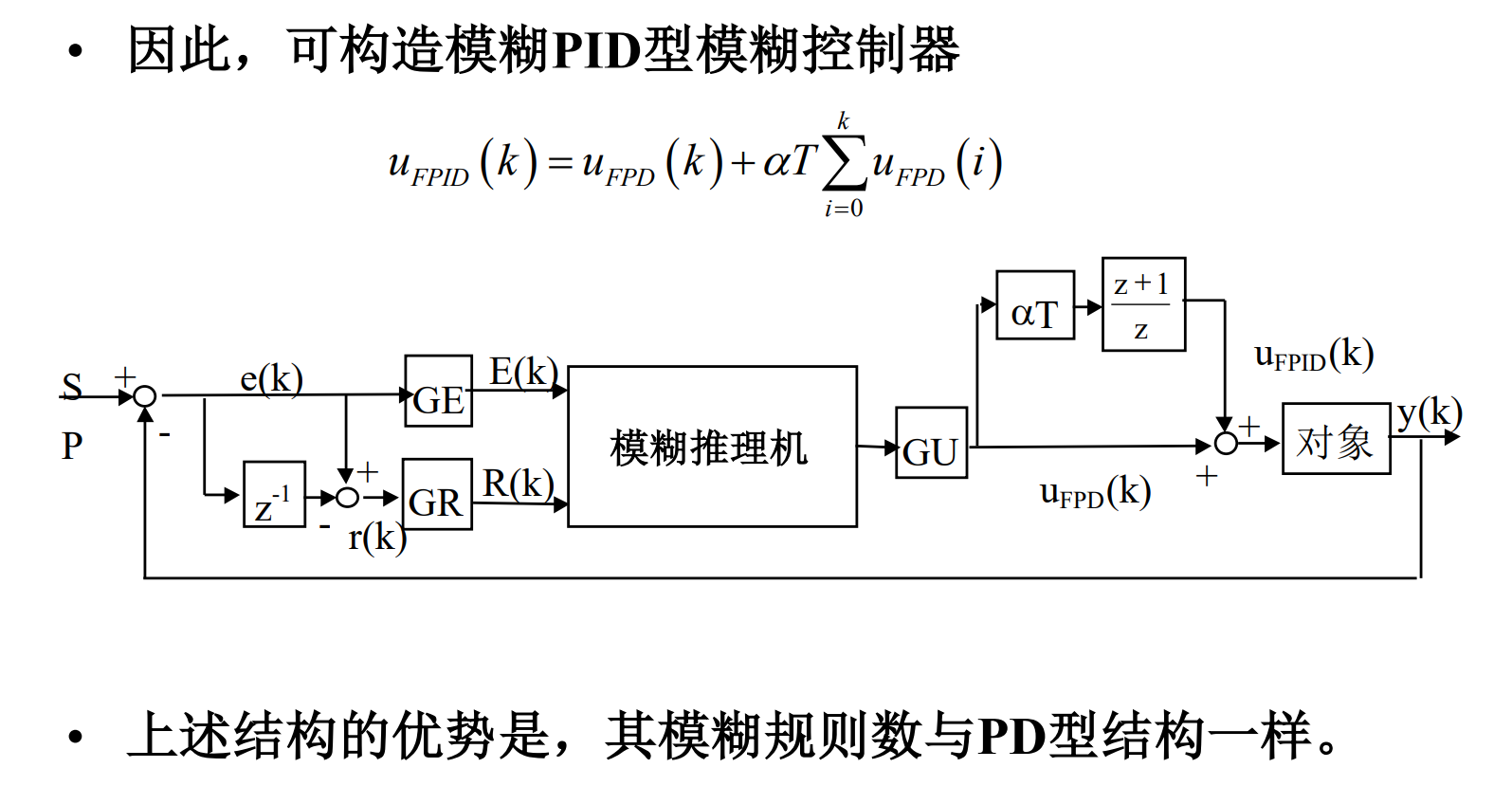


观察其输入如下：

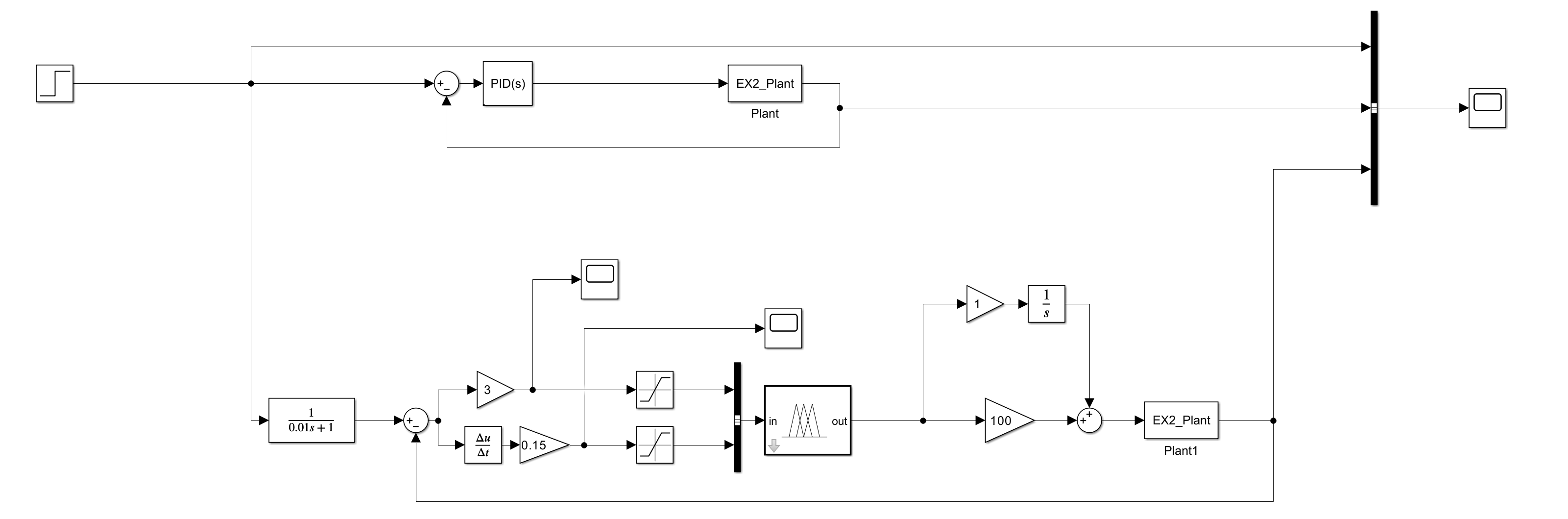


### FuzzyPID控制器

为减少不必要的、重复设置的模糊规则采用如下的模糊PID控制器构型



在Simulink中搭建控制系统如下：



值得强调的是：在模糊控制输出后进行运算再输入被控系统非常容易造成仿真系统判断模型发散或者陷入死循环。在模糊模型前设制saturation模块可以部分解决这一问题，但在调试过程中仍然非常频繁地出现系统陷入死循环的情况，故在此仅做模糊PID控制的说明和展示，不复现调节过程了。

主要关注FuzzyPID控制器控制系统进入稳态后的表现：



可以看出，模糊控制器在系统进入稳态后可以在大部分时间内将稳态偏差抑制在0.1%以内，但会经常性地产生最大约0.25%的波动，这可能主要是由于模糊控制器在输入均为0的原点附近平坦的控制规则造成的。