Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет систем управления и робототехники

Отчет по лабораторной работе №3 «Использование релейного, П и ПИД - регулятора»

по дисциплине «Введение в профессиональную деятельность»

Выполнили: студенты гр. R3137
Курчавый В.В.
Петров И.А.
Кирбаба Д.Д.
Нестеров И.А

Преподаватель: Перегудин А.А.,
ассистент фак. СУиР

Санкт-Петербург 2020

1. Цель работы

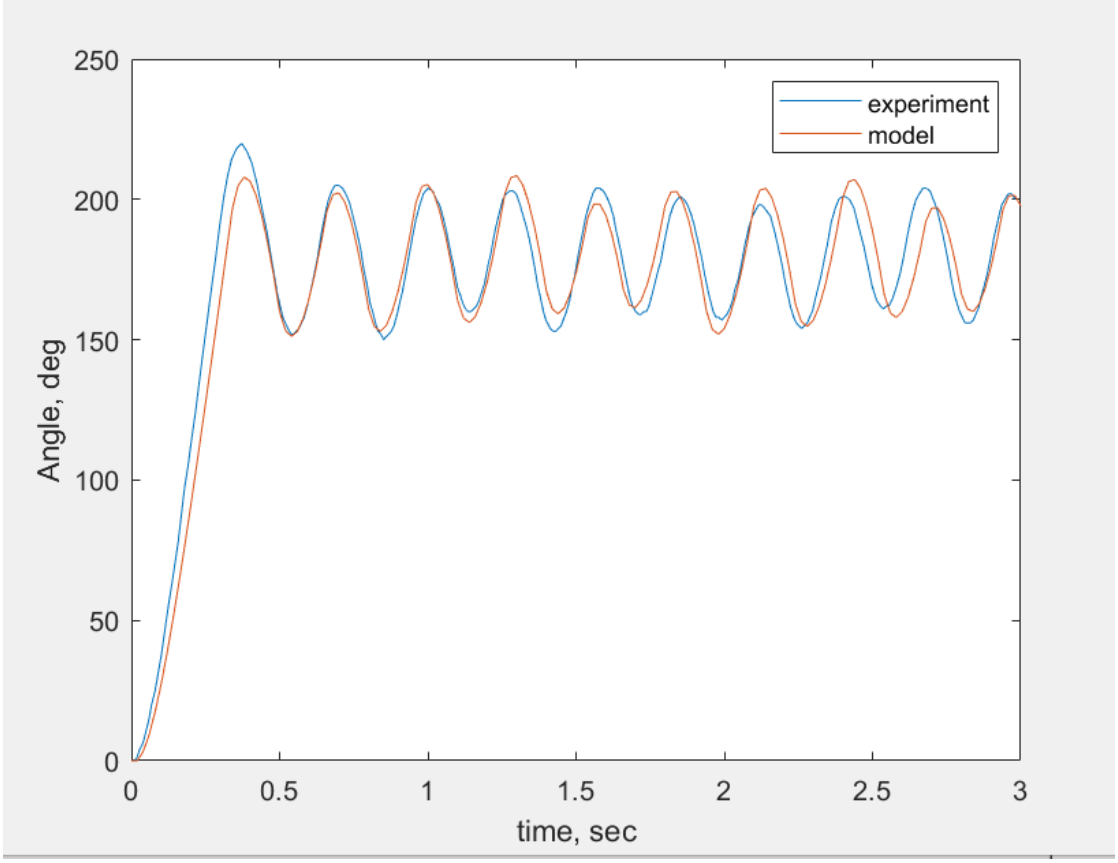
Сформировать релейный регулятор, пропорциональный регулятор и пропорционально-интегрально-дифференциальный регулятор. Изучить устройство управления на примере поворота мотора EV3 на 180 градусов.

2. Материалы работы

2.1 Результаты необходимых расчетов и построений

Wnls	14.644499694385809
Ke	0.48756937411725393
U_{max}	7.14020955025
J	0.002437632
L	0.0047
R	8.183683911882799

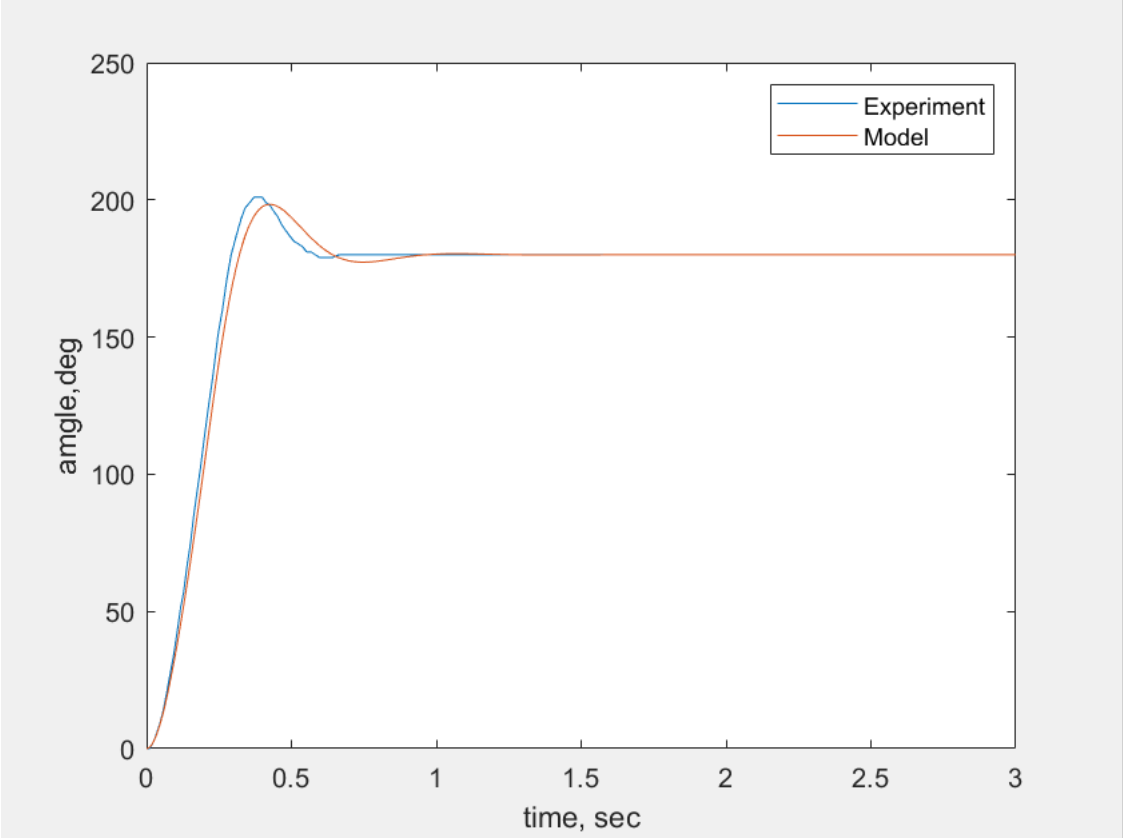
1. Релейный регулятор:



Графики модели и эксперимента расходятся, но незначительно. Амплитуда колебаний в эксперименте больше, так как двигатель ev3 подает управляющий сигнал один раз за время одной итерации цикла while (дискретно), а в Simulink управляющий сигнал подается чаще. Чтобы графики сходились, необходимо добавить в модель момент сил (Moth = 0.023) и интегратор, который преобразует значение угловой скорости вала в угол поворота, заменить на дискретный интегратор (sample time = 0.02).

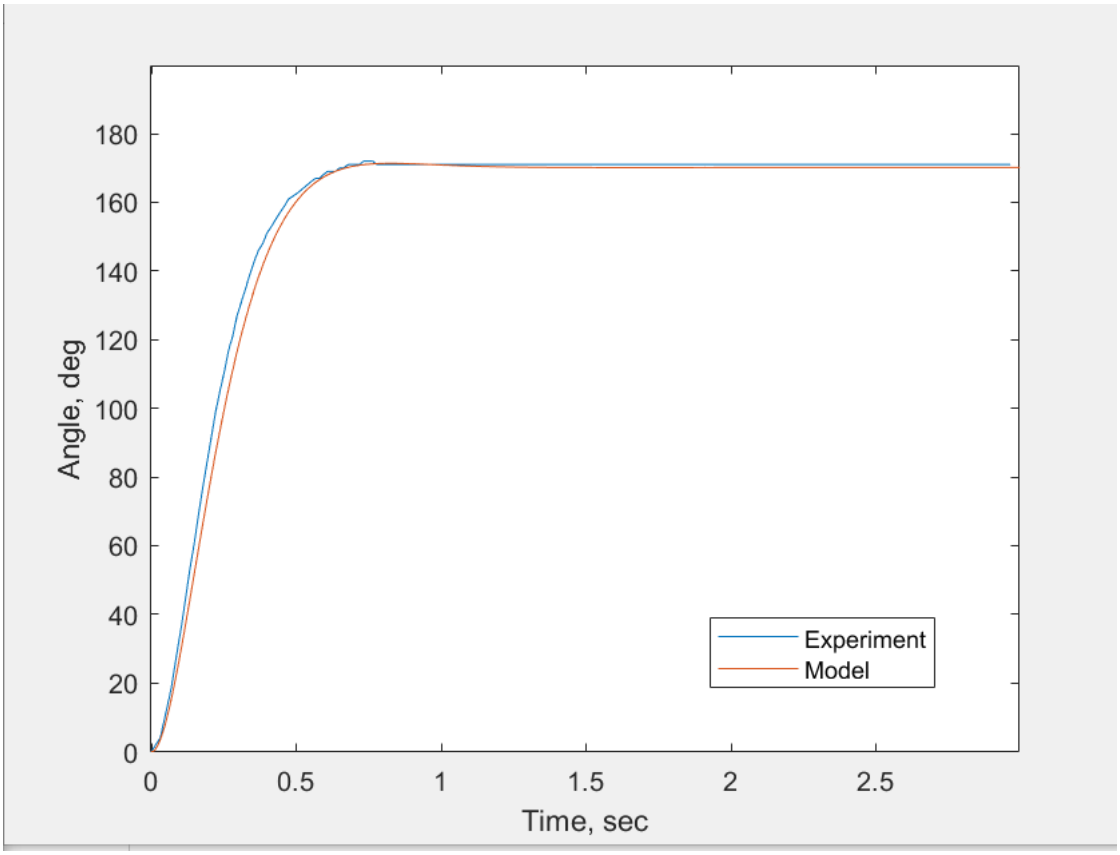
Установившаяся ошибка	20°
Максимальный угол отклонения	219°
Перерегулирование	9.5%

2. П – регулятор



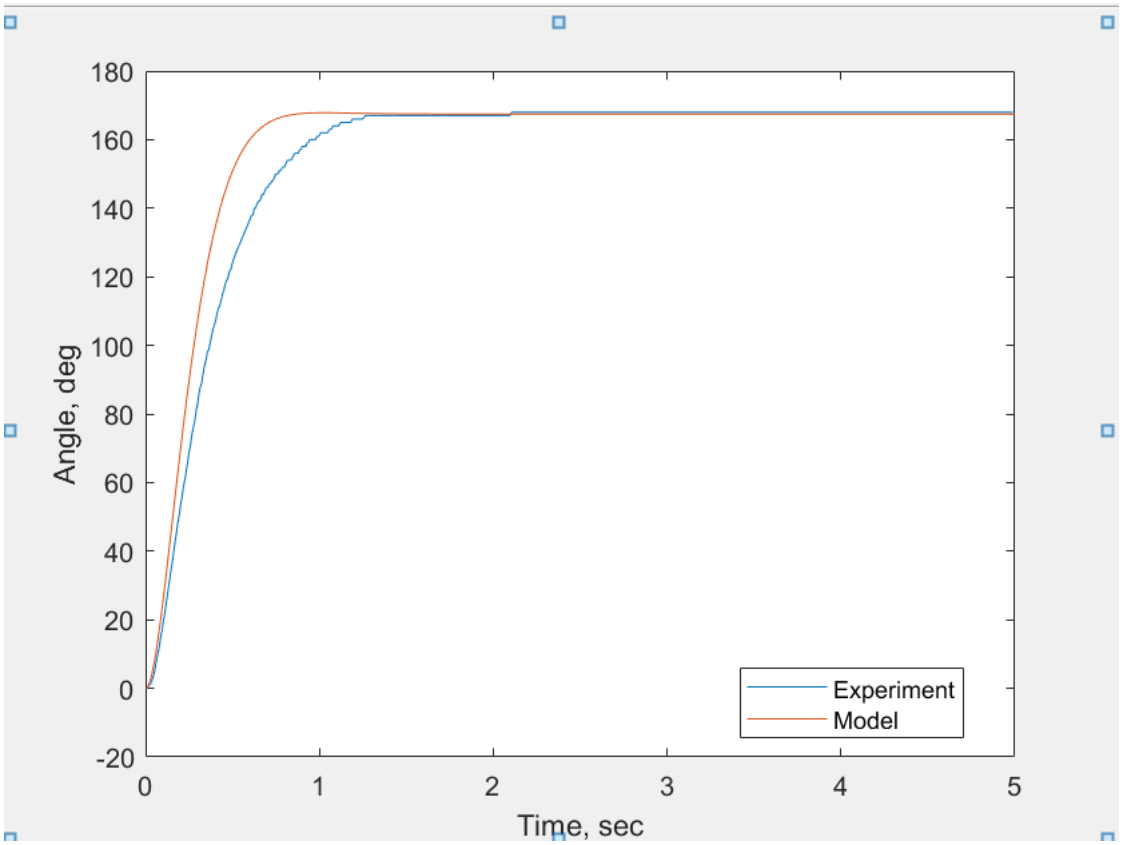
Графики эксперимента и модели совпадают Kn модели высчитывается по формуле `Kn = Kn_e*U_max*180/100/pi;`

Установившаяся ошибка	**1°**
Максимальный угол отклонения	201°
Перерегулирование	11,66%
Kn эксперимента	1.3
Kn модели	5.31
Время переходного процесса	0.52 с



Установившаяся ошибка в эксперименте и двигателя идентичные, так как в модели учитывается момент сил трения (Moth = 0.021).

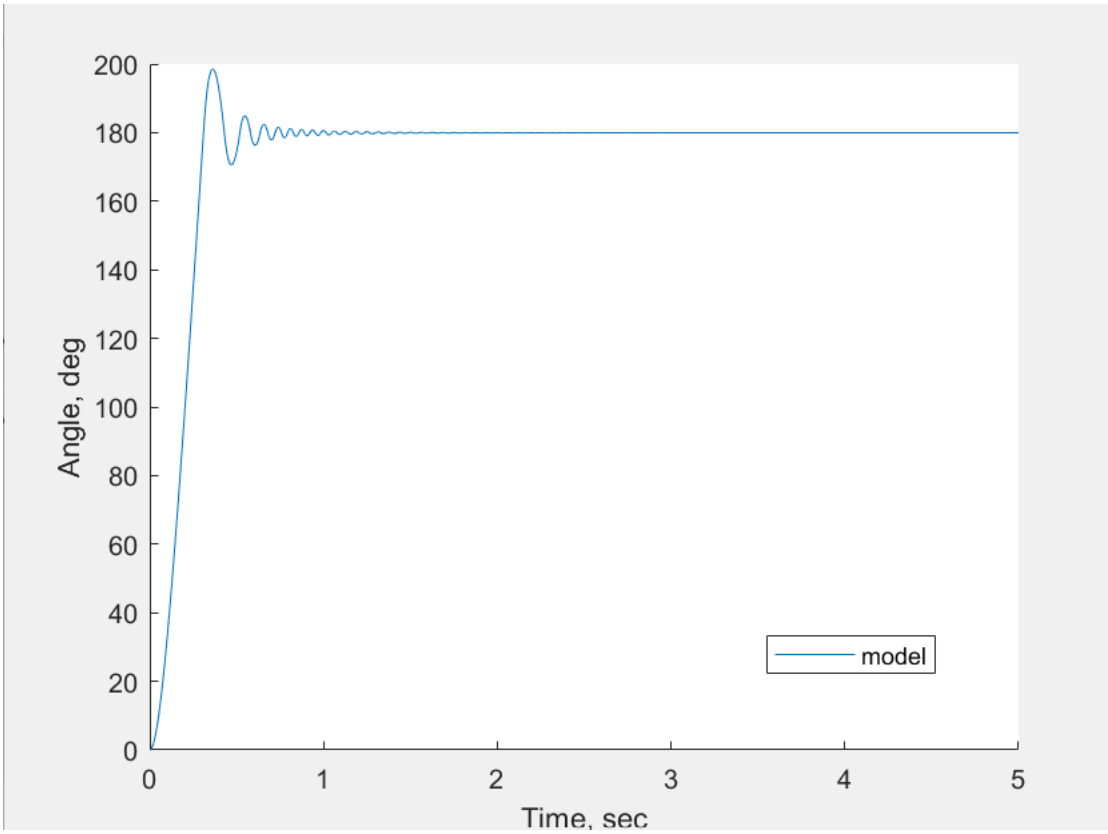
Установившаяся ошибка	9°
Максимальный угол отклонения	172°
Перерегулирование	0.58%
Kп эксперимента	0.5
Время переходного процесса	0.59 с
Kп модели	2.045



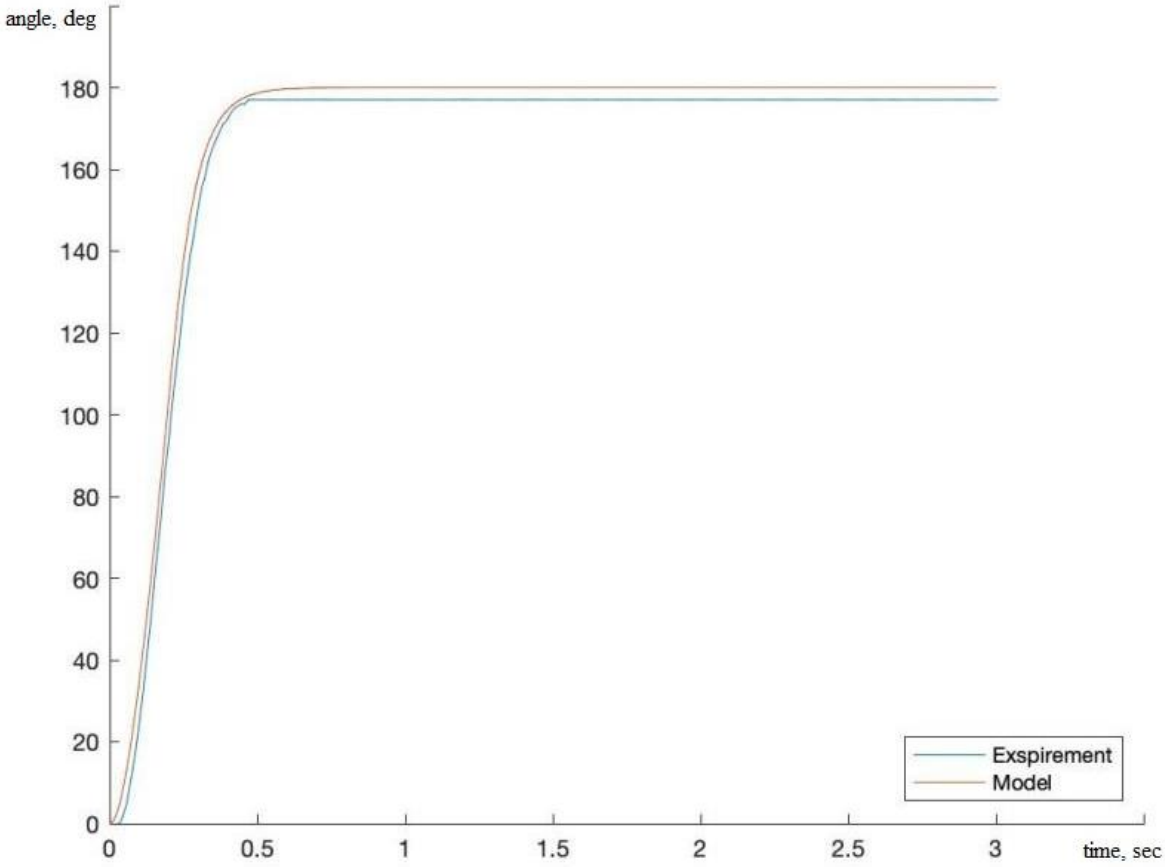
Установившаяся ошибка в эксперименте и двигателя идентичные, так как в модели учитывается момент сил трения (Moth = 0.021).

Установившаяся ошибка	12°
Максимальный угол отклонения	168°
Перерегулирование	0.0%
Kn	0.45

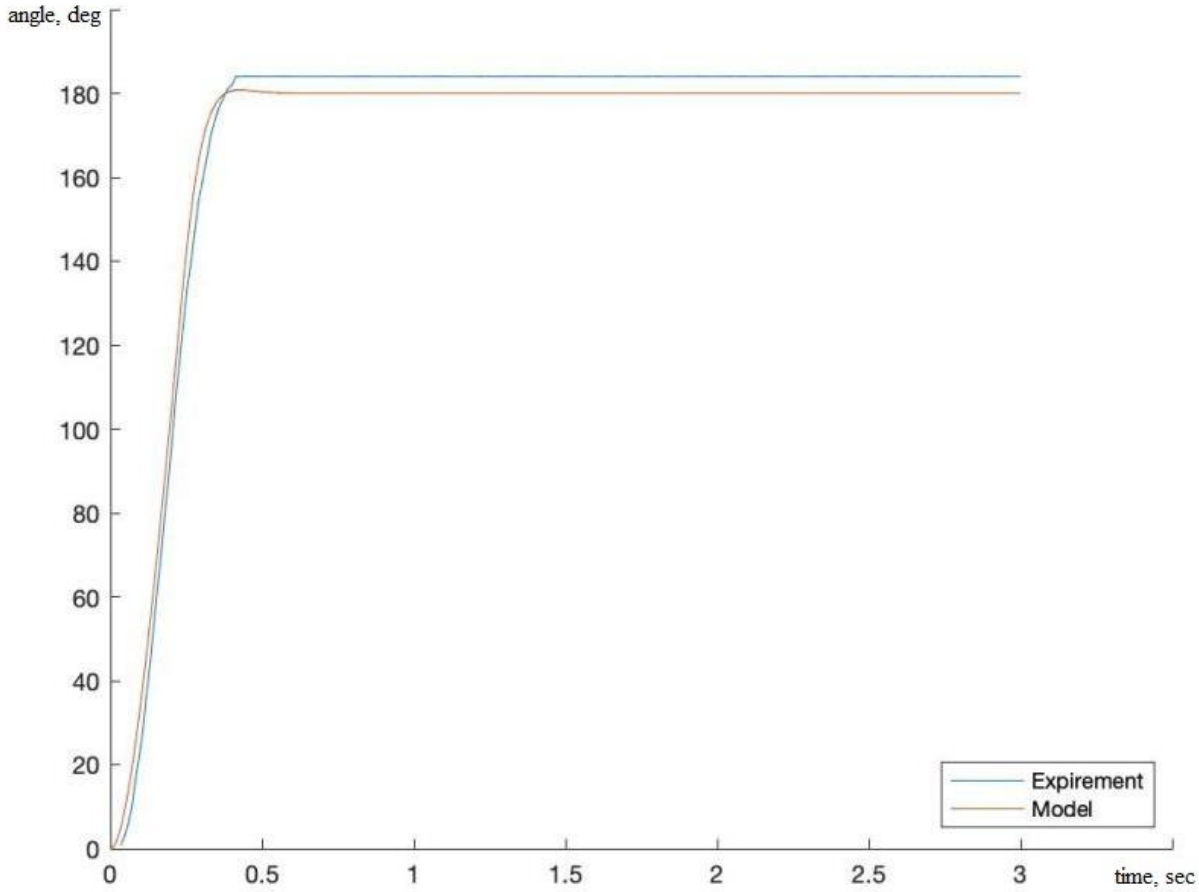
При большом коэффициенте Kn увеличивается перерегулирование и возникают затухающие колебания. К примеру, при Kn модели = 400:



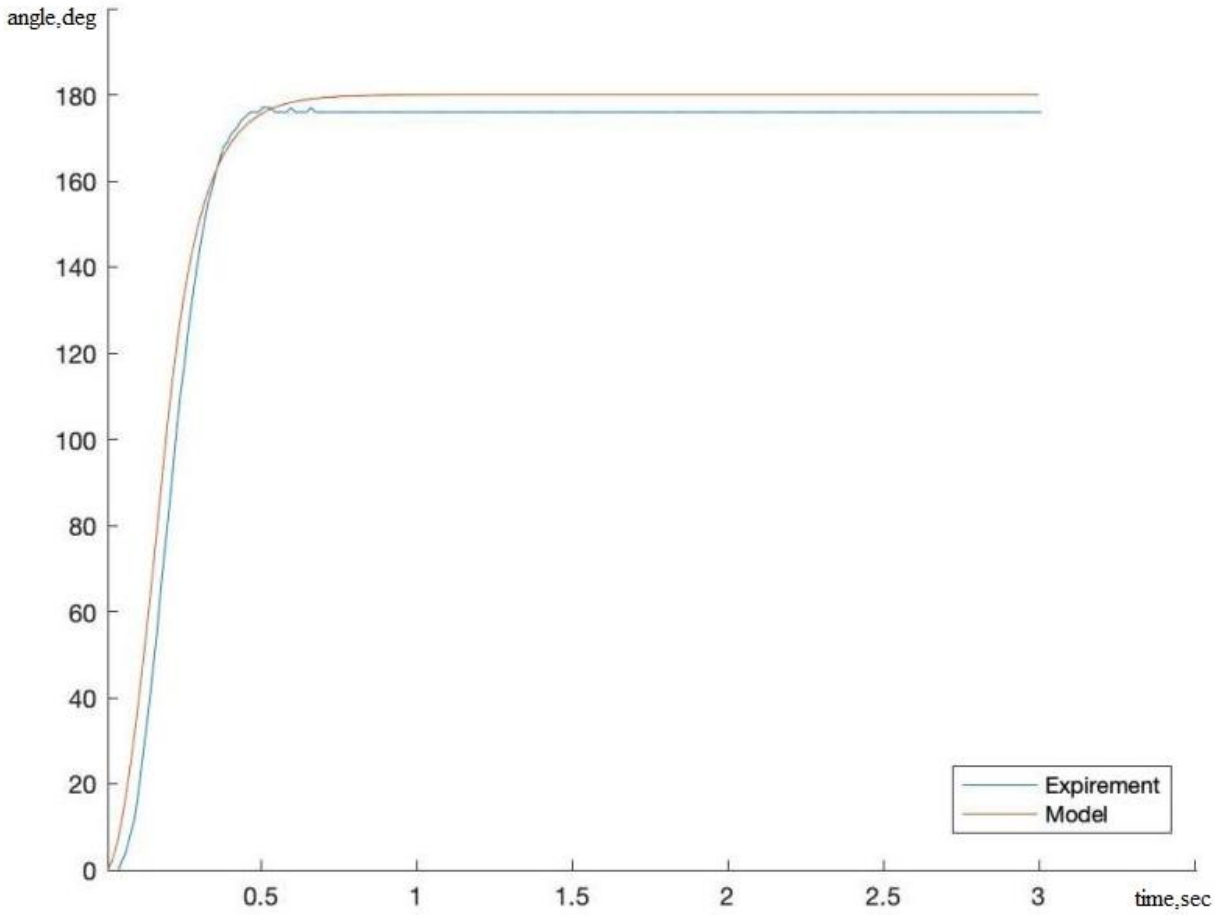
3. ПИД - регулятор



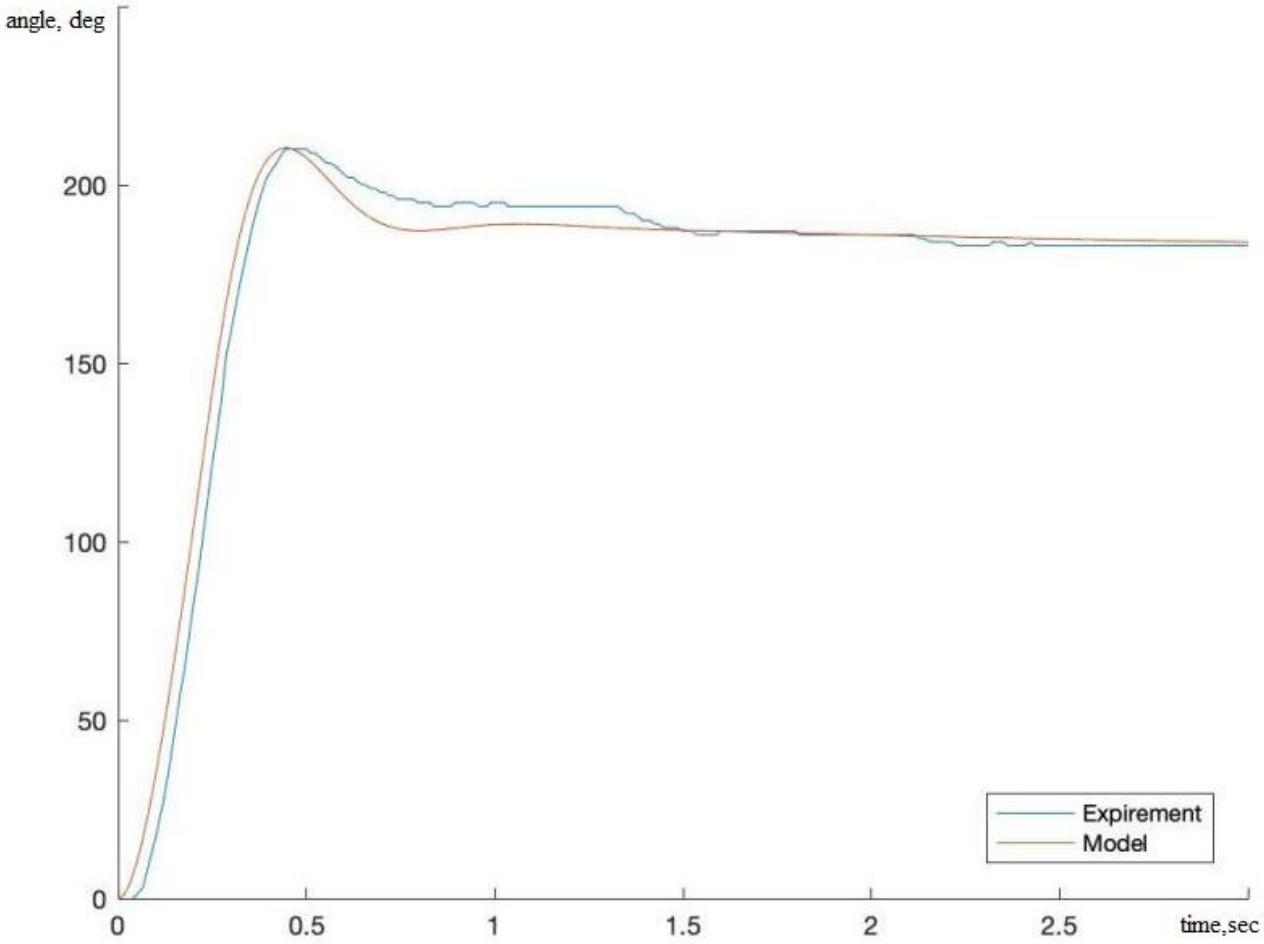
Установившаяся ошибка	3°
Максимальный угол отклонения	177°
Перерегулирование	0.0%
Kn	20
Kd experiment	0.5
Kd model	1.5



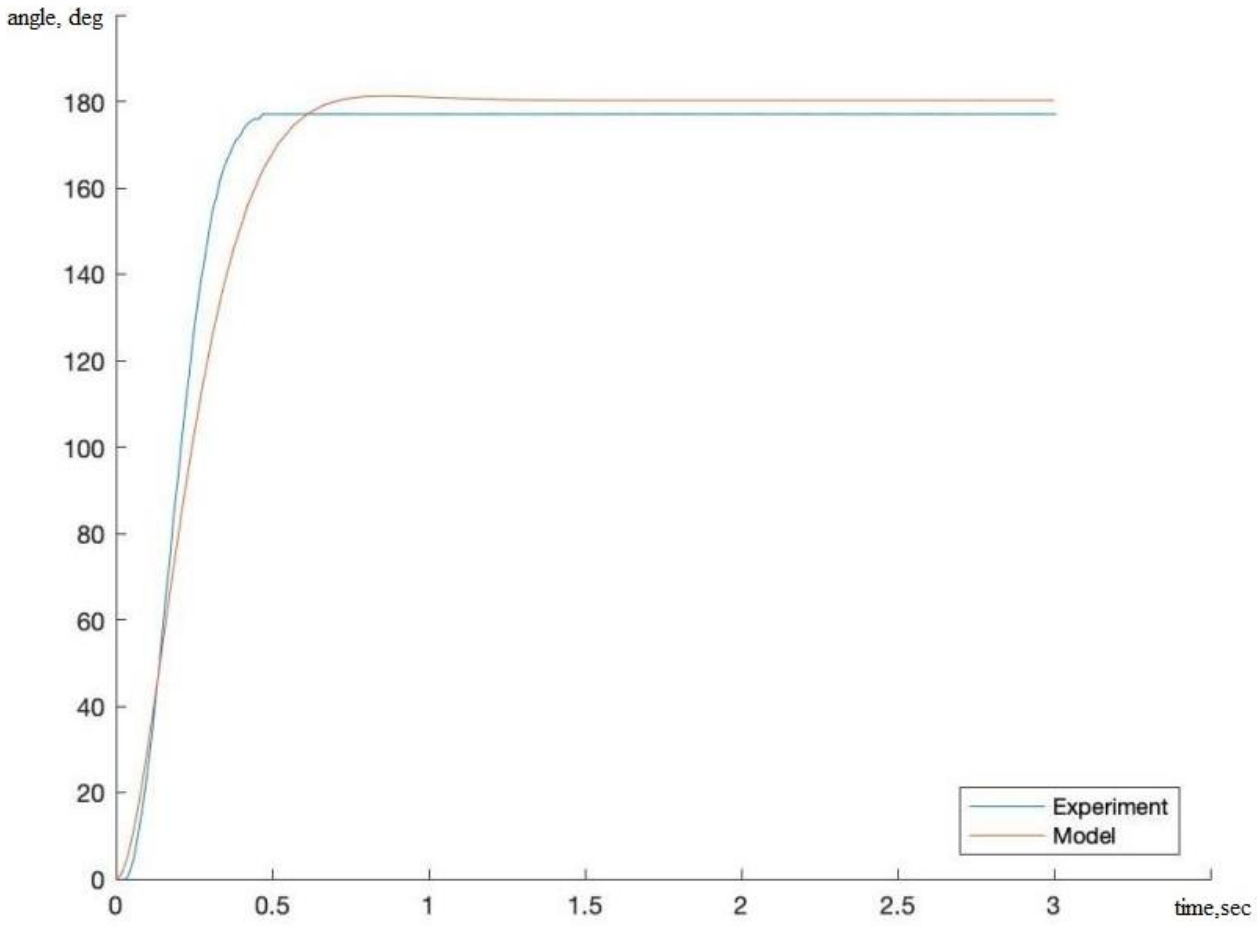
Установившаяся ошибка	3°
Максимальный угол отклонения	177°
Перерегулирование	0.4%
Kn	20
Kd experiment	1
Kd model	1



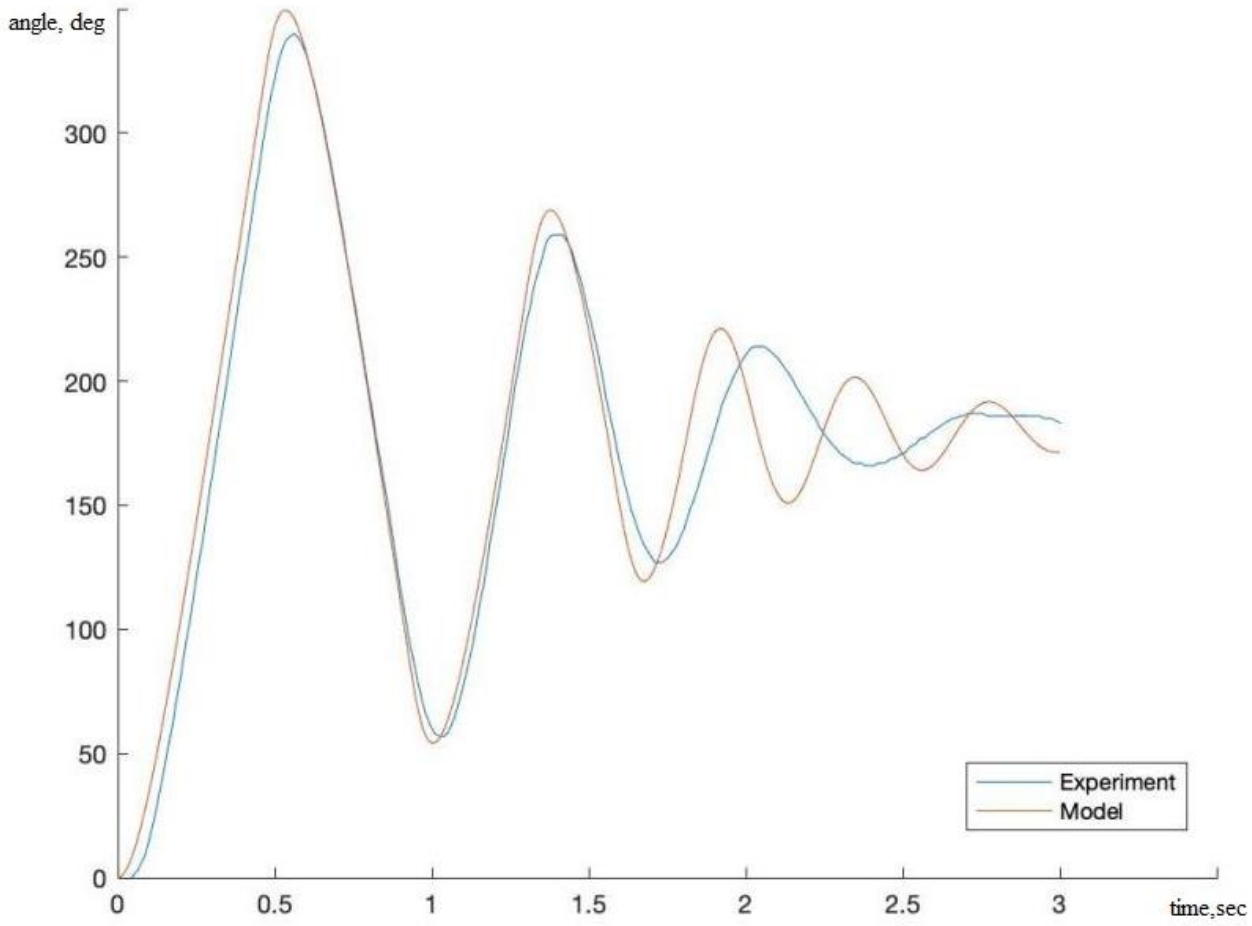
Установившаяся ошибка	4°
Максимальный угол отклонения	176°
Перерегулирование	0.35%
Kn	20
Kd experiment	2.5
Kd model	2



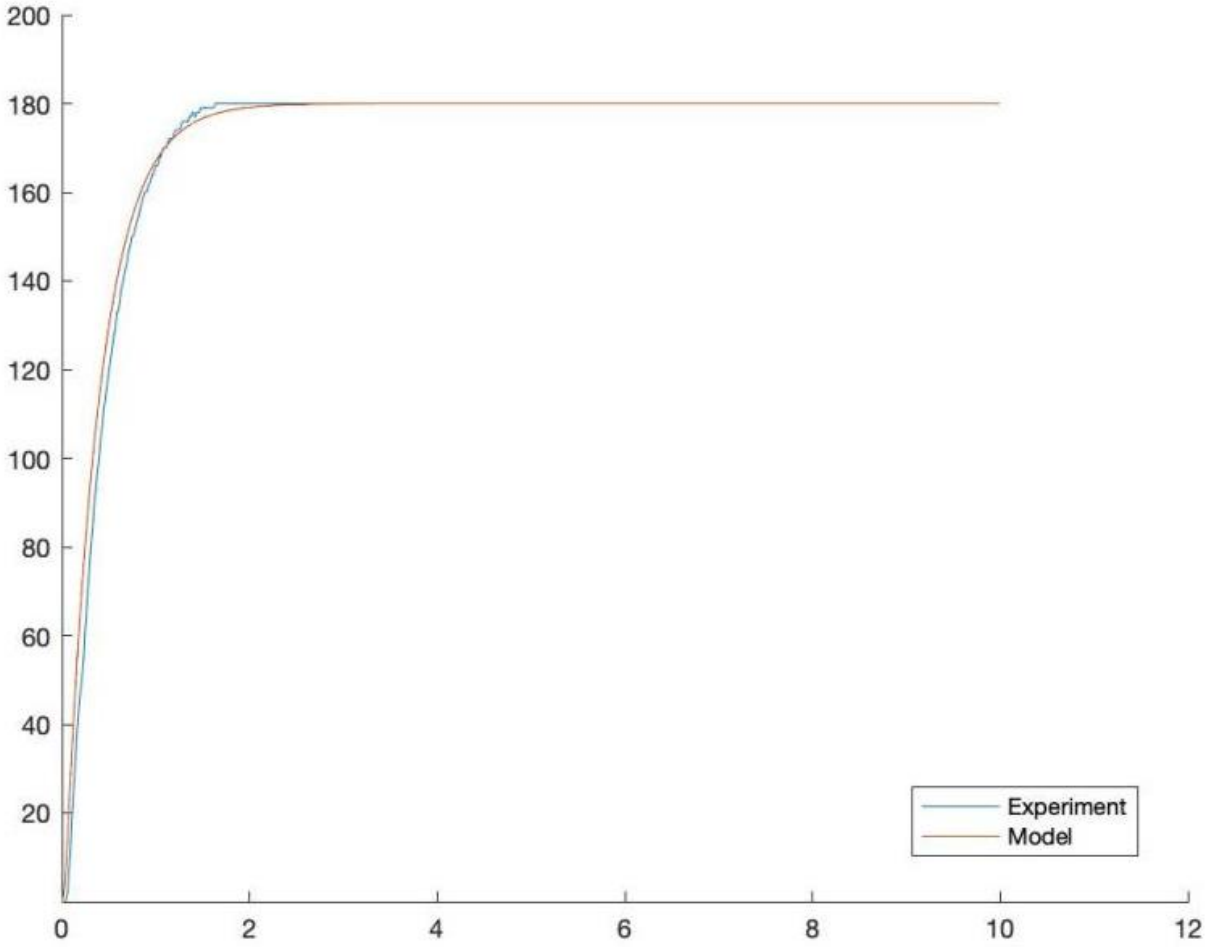
Установившаяся ошибка	32°
Максимальный угол отклонения	212°
Перерегулирование	13%
Kn	5
Ki experiment	0.01
Ki model	2



Установившаяся ошибка	2°
Максимальный угол отклонения	178°
Перерегулирование	0%
Kn	2
Ki experiment	0.5
Ki model	0.01

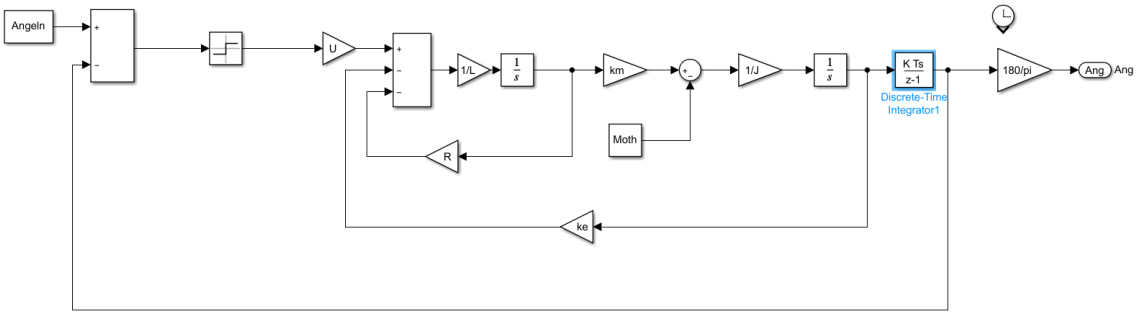


Установившаяся ошибка	6°
Максимальный угол отклонения	343°
Перерегулирование	190%
Kn	10
Ki experiment	0.1
Ki model	70

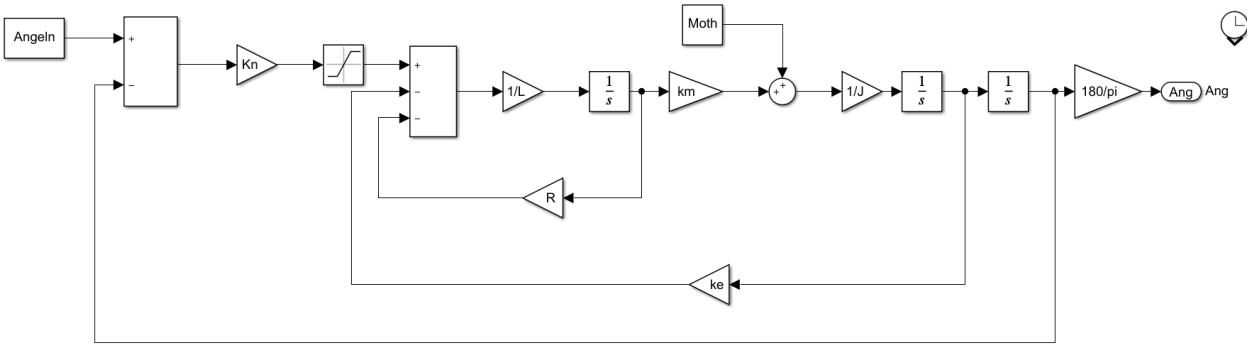


Установившаяся ошибка	0°
Максимальный угол отклонения	181°
Перерегулирование	0.09%
Kn experiment	1
Ki experiment	0.001
Kd experiment	21
Kn model	20
Ki model	0.001
Kd model	7

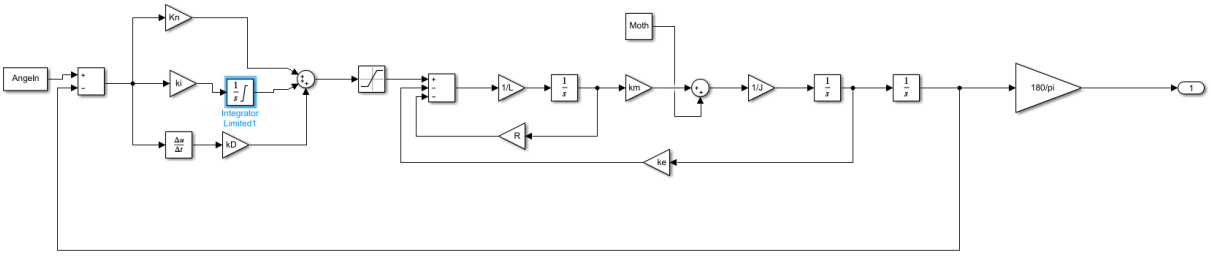
Модель для релейного регулятора:



Модель для П-регулятора:



Модель для ПИД-регулятора:



Код python ПИД-регулятор:

```
def degree_rotation(angle):
    motorA = LargeMotor('outA')
    timeStart = time.time()
    motorA.position = 0
    timeNow = time.time() - timeStart
    data = open('data_for_voltage_' + "kp=1.0;ki=0.001;kd=21.0" + '_.txt', 'w')
    kp = 1.0
    ki = 0.001
    kd = 21.0
    e = 0
    integral = 0
    derivative = 0
    tp = time.time()
    tpend = 0
    tetamax = 0
    try:
        while (timeNow < 10):
            tp = time.time()
            preve = e
            e = angle - motorA.position
            integral += e
            derivative = e - preve

            if (abs(angle - motorA.position) < 0.05*angle) and (tpend == 0):
                tpend = tp - timeStart
            if tpend != 0 and not(abs(angle - motorA.position) < 0.05*angle):
                tpend = 0

            if  kp*e + ki*integral + kd*derivative > 100:
                voltage = 100
            elif kp*e + ki*integral + kd*derivative < -100:
                voltage = -100
            else: voltage = kp*e + ki*integral + kd*derivative

            motorA.run_direct(duty_cycle_sp = voltage)
            timeNow = time.time() - timeStart
            data.write(str(motorA.position))
            data.write('\t' + str(round(timeNow,3)) + '\n')

            if tetamax < motorA.position:
                tetamax = motorA.position
            teta = motorA.position
    finally:
```

```
    finally:
        data.write("Est" + str(angle - teta) + "\n")
        data.write("Tetamax" + str(tetamax) + "\n")
        data.write("Perereg" + str((tetamax - teta)/(teta)*100) + "%\n")
        data.write("tp = " + str(tpend))
        data.close()
        motorA.stop(stop_action = 'brake')
        time.sleep(3)

degree_rotation(180)
```

Код-python для релейного регулятора:

```python
def degree_rotation(angle):

    motorA = LargeMotor('outA')
    timeStart = time.time()
    motorA.position = 0
    timeNow = time.time() - timeStart
    tp = time.time()
    tpend = 0
    tetamax = 0

    data = open('degree_rotation_' + "180" + '_.txt', 'w')

    while (timeNow < 10):
        tp = time.time()
        if (abs(angle - motorA.position) < 0.05*angle) and (tpend == 0):
            tpend = tp
        if tpend != 0 and not(abs(angle - motorA.position) < 0.05*angle):
            tpend = 0
        if motorA.position < angle:
            voltage = 100
        elif motorA.position == angle:
            voltage = 0
        elif motorA.position > angle:
            voltage = -100
        motorA.run_direct(duty_cycle_sp = voltage)
        timeNow = time.time() - timeStart
        data.write(str(motorA.position))
        data.write('\t' + str(round(timeNow,3)) + '\n')
        if tetamax < motorA.position:
            tetamax = motorA.position


    data.write("Est" + str(angle - motorA.position) + "\n")
    data.write("Maxteta" + str(tetamax) + "\n")
    data.write("Pererg" + str((tetamax - motorA.position)/motorA.position*100) + "%")
    data.write("tp = " + str(tpend))
    data.close()
    motorA.stop(stop_action = 'brake')


degree_rotation(180)
```

Расчет Umax:

U = wnls*ke = 14.644499694385809*0.48756937411725393

Код Scilab для расчета wnls:

```
results = read ("C:\Users\vital\itmo\professionalActivity\lab2\data_for_voltage_-100_.txt", -1, 2) // Считывание файла
angle = results(:,1)*%pi/180 // Создание матрицы с данными об угле поворота
time = results(:,2) // Создание матрицы с данными об времени
plot2d(time, angle, 2) // Построение графика на основе считанных данных
aim=[time,angle] // Формирование матрицы для аппроксимации
aim=aim' // Транспонирование матрицы (замена столбцов на строки)
deff('e=func(k,z)','e=z(2)-k(1)*(z(1)-k(2)*(1-exp(-z(1)/k(2))))') // Объявление функции, чьи коэффициенты будут
определяться при аппроксимации
att=[-15;0.06] // Для отрицательных указать отрицательное значение на первой позиции
[koeffs,errs] = datafit(func,aim,att) //
Wnls = koeffs(1)
Tm = koeffs(2)
j = 0.0023
Mst = j*Wnls/Tm
model=Wnls*(time-Tm*(1-exp(-time/Tm)))
plot2d(time,model,3)
legend('Experiment','$\theta(t)=\omega_{nls}t-\omega_{nls}T_m+ \omega_{nls}T_m\,exp\bigl(-
\frac{t}{T_m}\bigr)$','Model',2)
```

Код Matlab для построения графиков:

```
Res = dlmread('pireg180_.txt');
Res = Res';
Angle = Res(1,:);
Time = Res(2,:);
plot(Time, Angle);
Angeln = pi;
ke = 0.48756937411725393;
km = ke;
J = 0.002437632;
L = 0.0047;
Kn = 1.25;
R = 8.183683911882799;
U = 14.6444996943858090.48756937411725393;
Kn = Kn*U*180/100/pi;
TimeM = out.yout{1}.Values.Time;
AngleM = out.yout{1}.Values.Data;
hold on
plot(TimeM,AngleM);
```

3. Выводы:

Мы построили релейный регулятор, П-регулятор и ПИД-регулятор. Изучили устройство управления на примере поворота мотора EV3 на 180 градусов. Построили графики функций для эксперимента и модели. При сходящихся графиках имеются разные коэффициенты, так как, считая производную и интеграл в эксперименте мы не учитывали dt и напряжение в эксперименте мы подаем в %, а Matlab работает в системе Си.

При увеличении пропорционального коэффициента время переходного процесса уменьшается. При умеренном увеличении коэффициента уменьшается установившаяся ошибка, но она никогда не будет равна нулю. При большом увеличении коэффициента возникают незатухающие колебания, вследствие чего увеличивается установившееся ошибка и увеличивается перерегулирование.

При увеличении интегральной составляющей уменьшается установившаяся ошибка, но увеличивается перерегулирование, время переходного процесса почти не меняется.

Дифференциальный коэффициент уменьшает перерегулирование и снижает возможные колебания системы.