Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет систем управления и робототехники

Отчет по лабораторной работе №4

«НАСТРОЙКА ПИД-регулятора»

по дисциплине «Введение в профессиональную деятельность»

Выполнили: студенты группы R3136

Мартынова А.А.

Мангутов Н.Р.

Шахтаров И.Р.

Корякина Е.Н.

Преподаватель: Перегудин А.А.

Санкт-Петербург

2019

1. Цель работы  
   Познакомиться с концепцией ПИД – регулятора. Получить опыт настройки его параметров, попытавшись решить определенную задачу управления для мобильного робота с дифференциальным приводом.
2. Результаты работы  
   2.1. Изучение работы П – регулятора в установке с одним ДПТ  
   Зафиксируем и пронаблюдаем за изменением установившейся ошибки перерегулирования σ и времени переходного процесса .

|  |  |  |  |
| --- | --- | --- | --- |
| *Kp* | *e, °* | *σ, %* | *tп,с* |
| 1 | 3 | 3,28 | 0,34 |
| 2 | 1 | 13,4 | 0,47 |
| 3 | 1 | 22,9 | 0,62 |
| 4 | 0 | 16,1 | 0,56 |
| 5 | 0 | 17,7 | 0,79 |

Из таблицы видно, что при увеличении *kp* установившаяся ошибка уменьшается, перерегулирование увеличивается, а время переходного процесса увеличивается.

|  |  |  |  |
| --- | --- | --- | --- |
|  | *e, °* | *σ, %* | *tп,с* |
| *kp*  ↑ | ↓ | ↑ | ↑ |

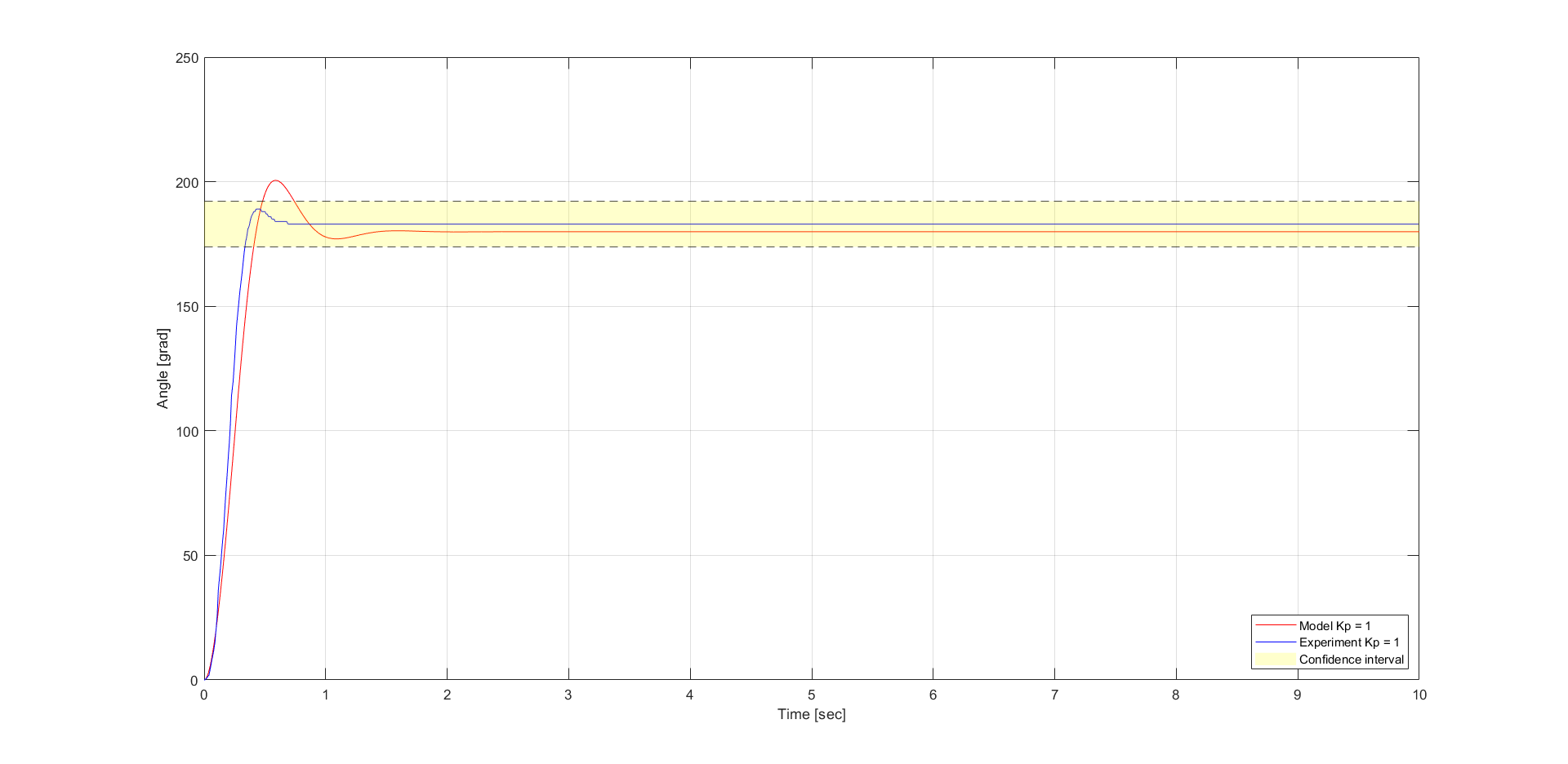


Рисунок 1. Зависимость угла от времени при kp = 1

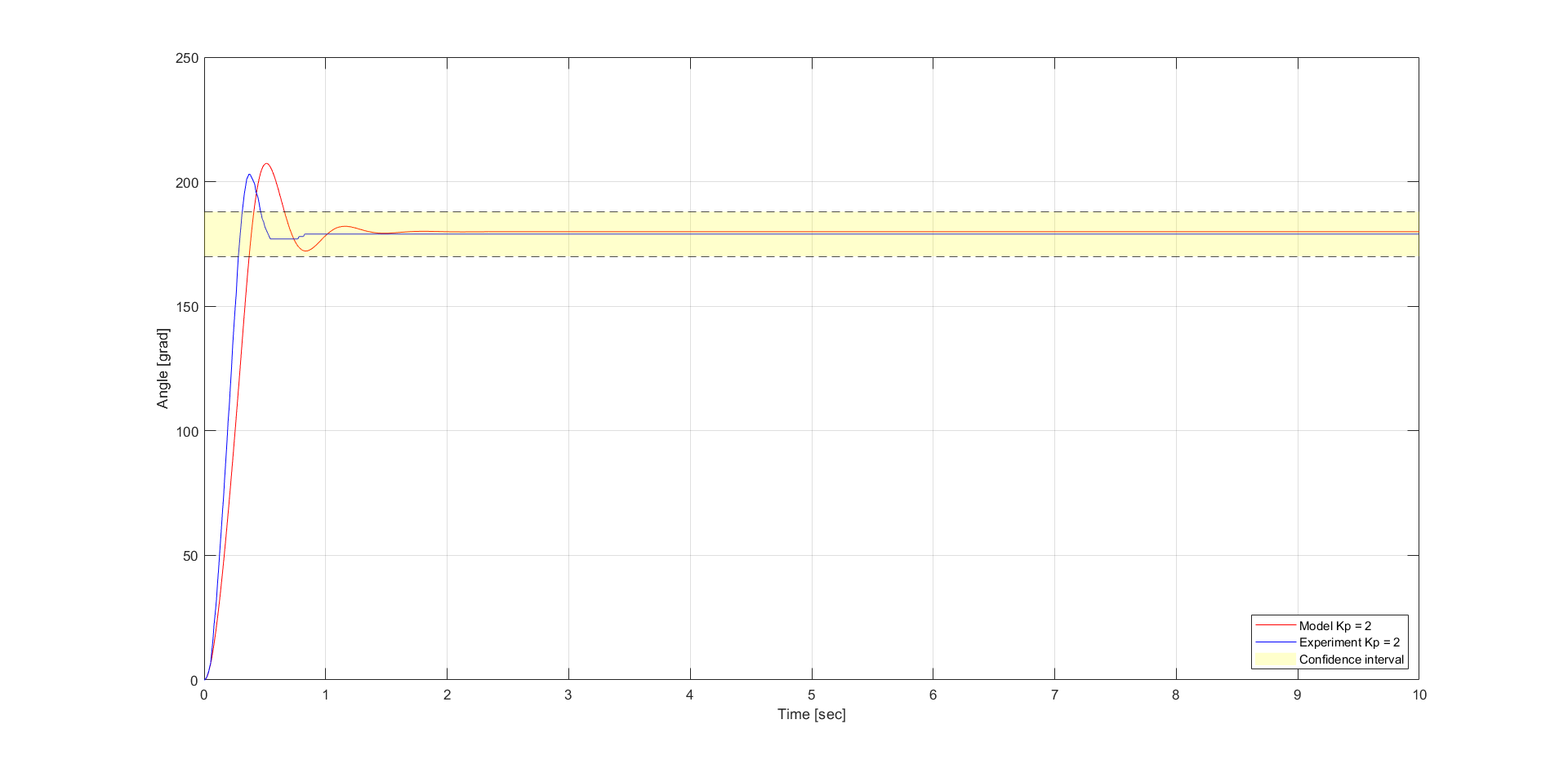


Рисунок 2. Зависимость угла от времени при kp = 2

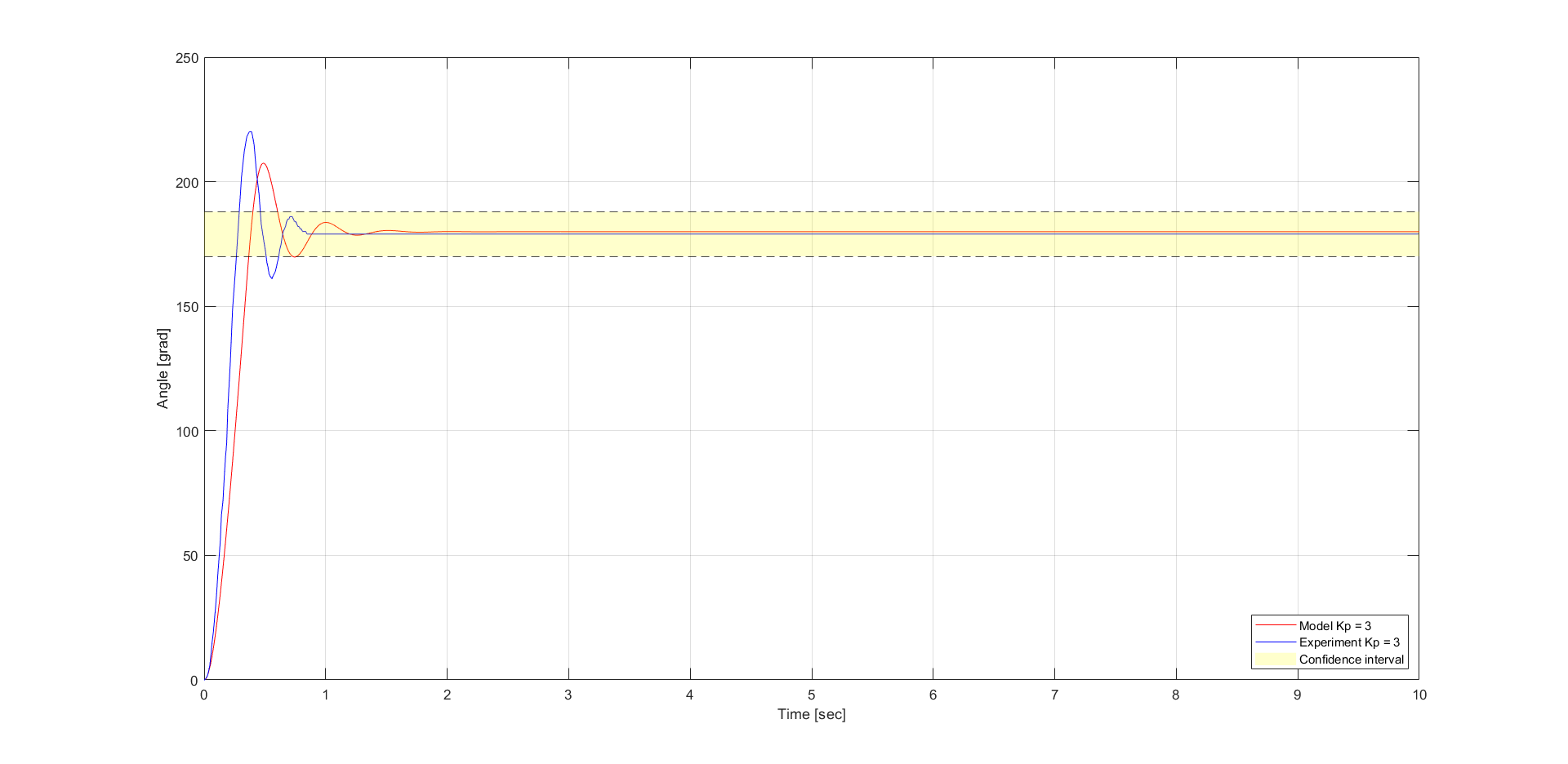


Рисунок 3. Зависимость угла от времени при kp = 3

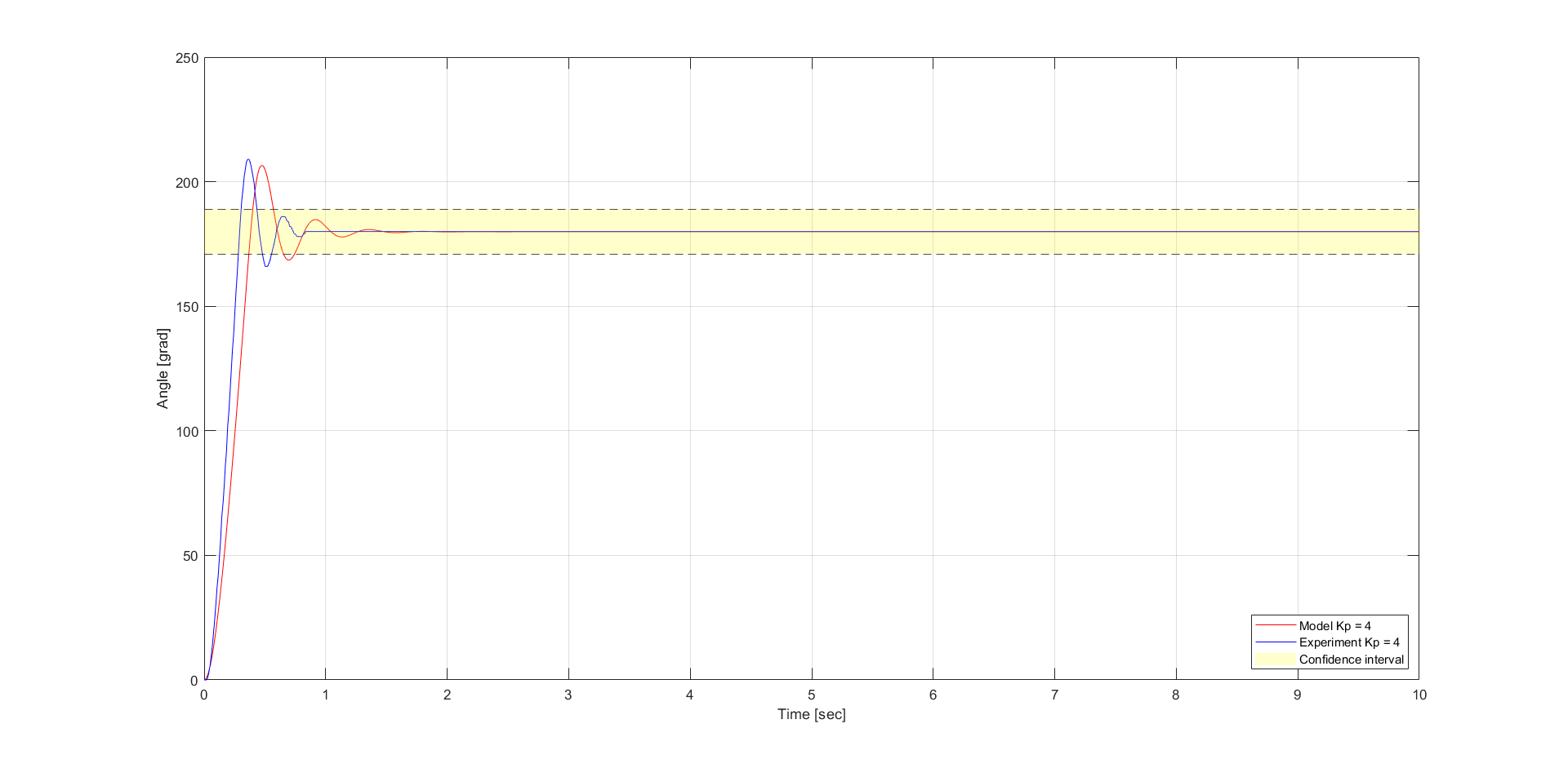


Рисунок 4. Зависимость угла от времени при kp = 4

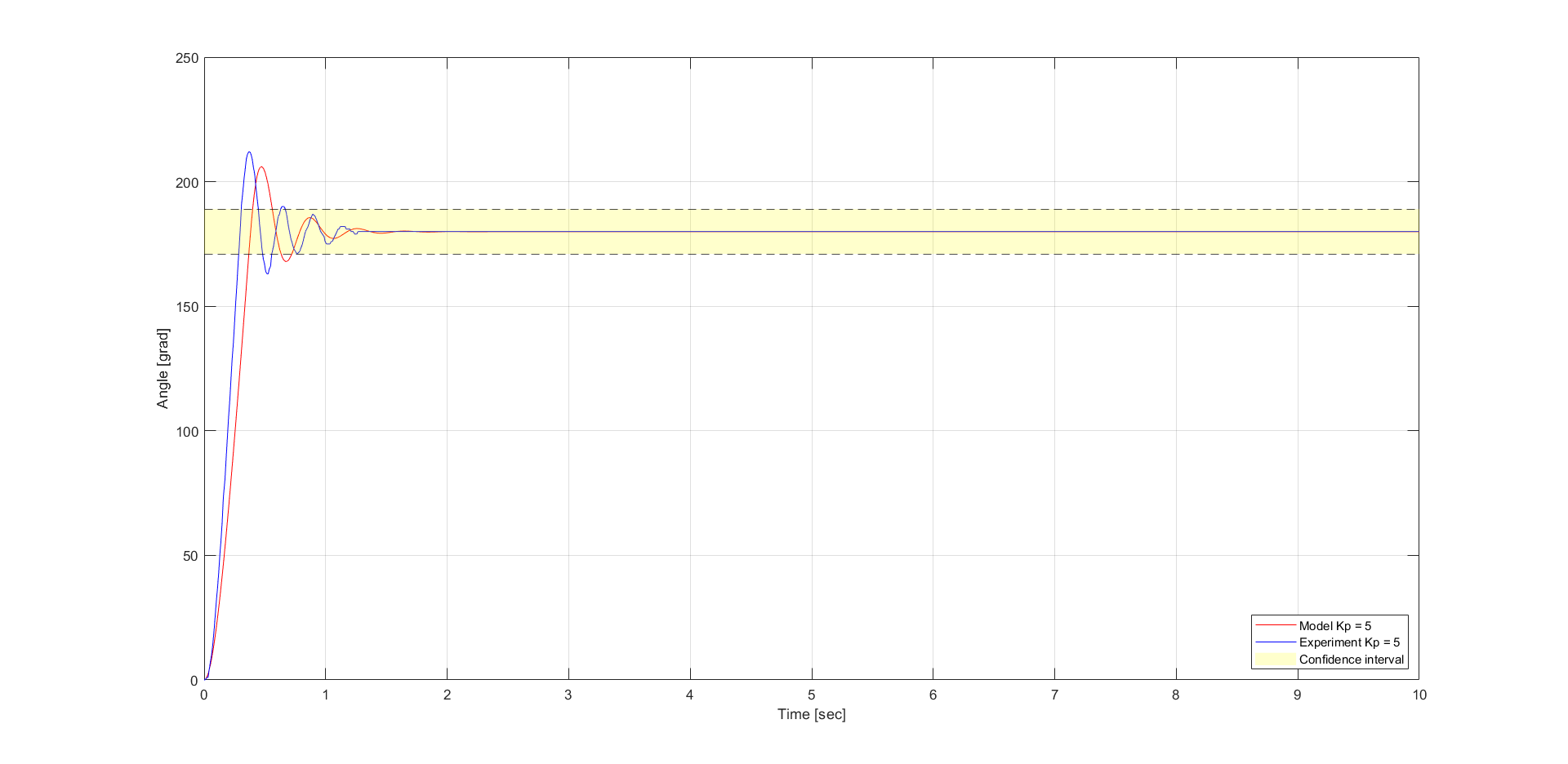


Рисунок 5. Зависимость угла от времени при kp = 5

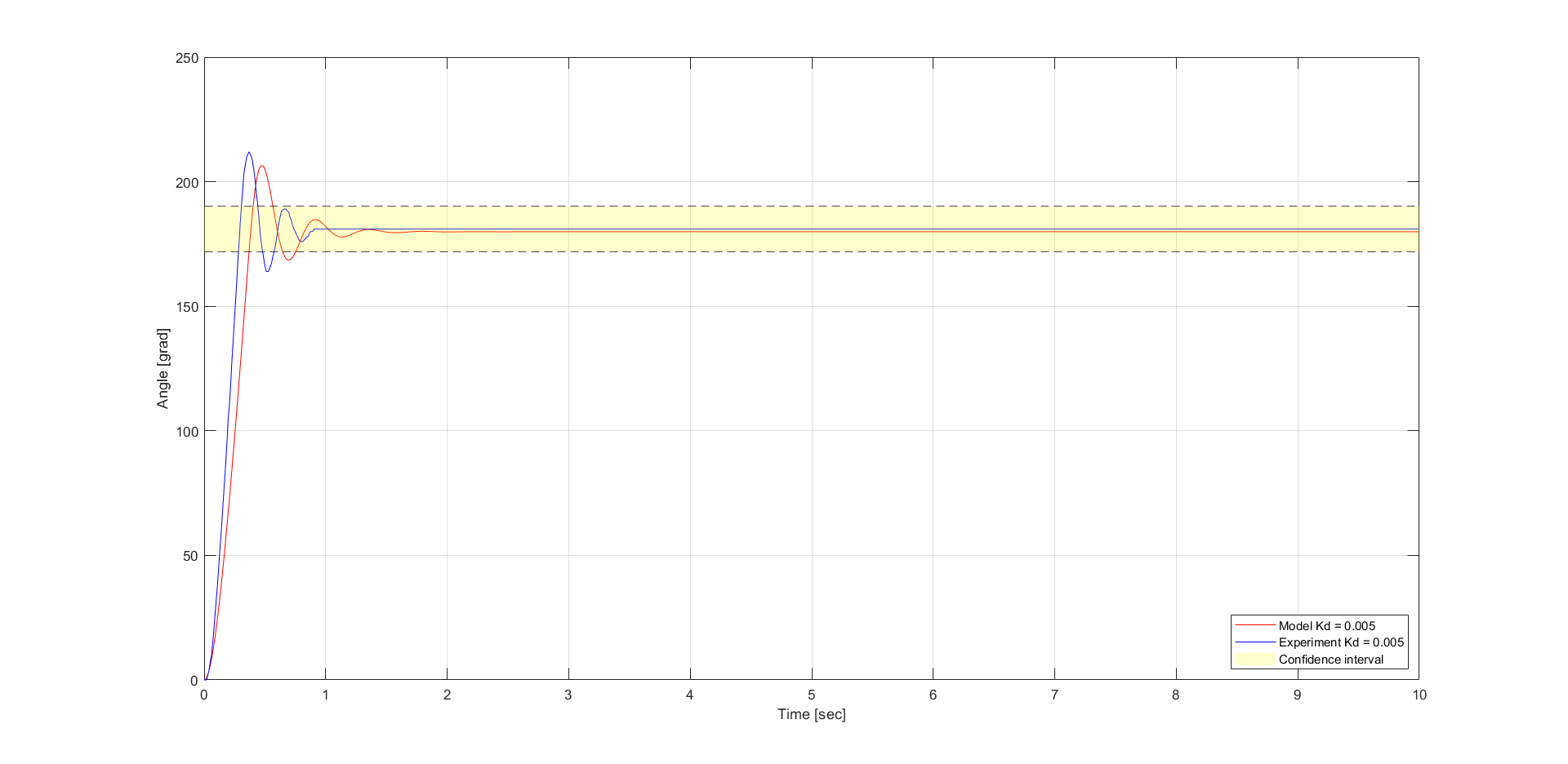
* 1. Изучение работы ПД – регулятора  
     Из предыдущего опыта *kp =4* оказался наиболее удачным (нулевая установившаяся ошибка за наименьшее время). Зафиксируем его и *ki* = *0* и посмотрим, как будут вести себя *e, σ* и *tп* при изменении *kd*.

Рисунок 6. Зависимость угла от времени при kd = 0.005

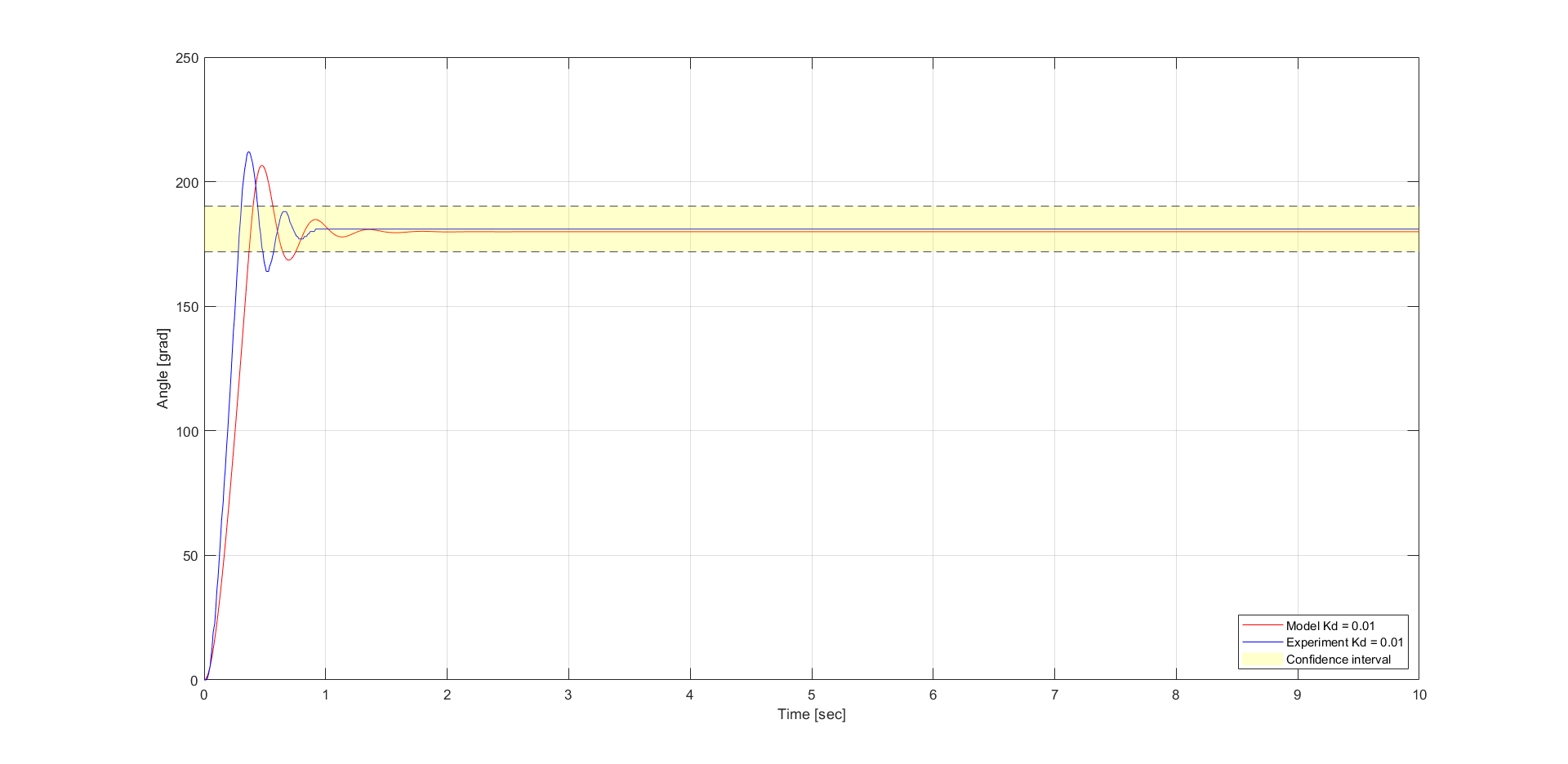


Рисунок 7. Зависимость угла от времени при kd = 0.01

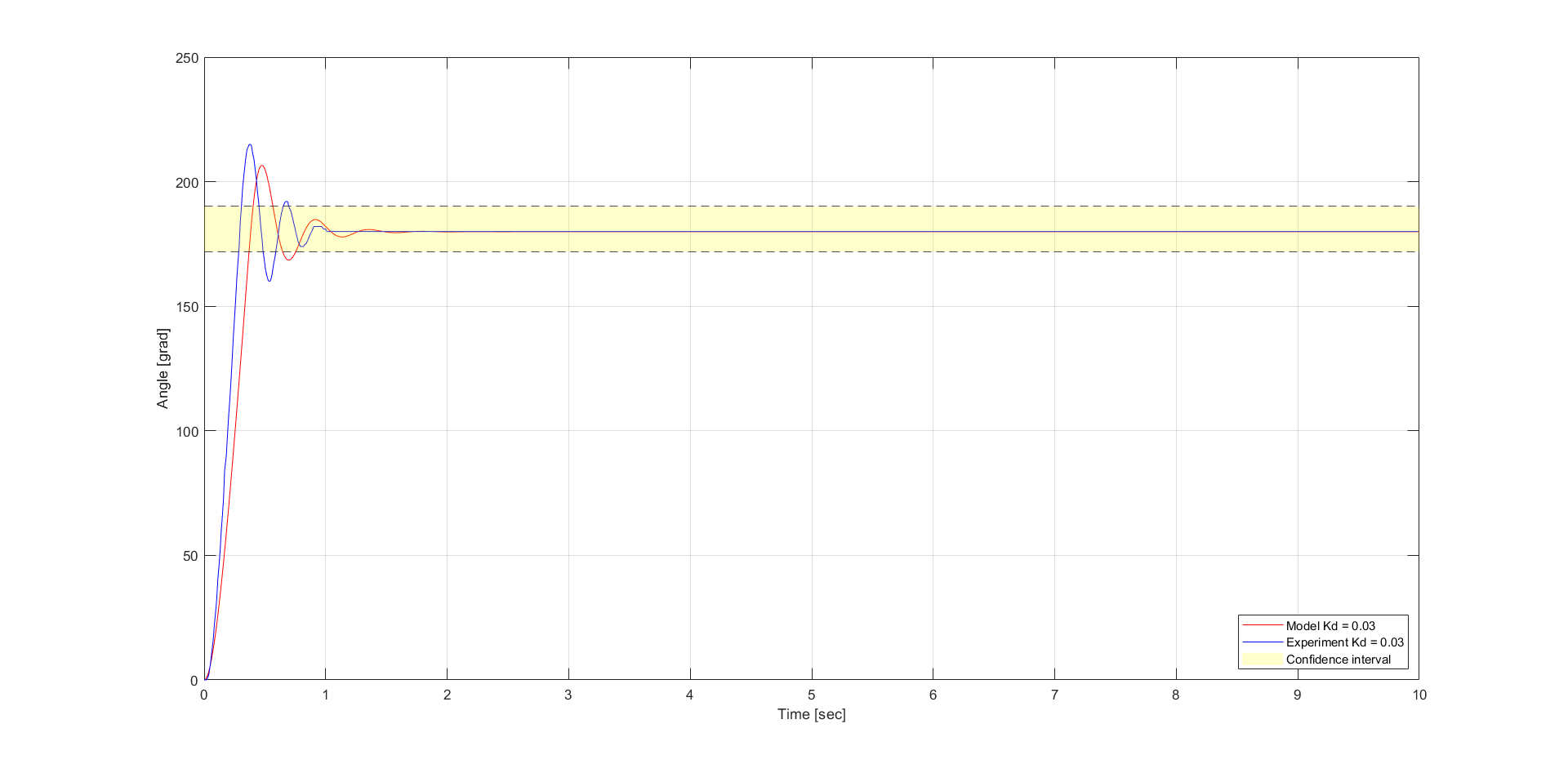


Рисунок 8. Зависимость угла от времени при kd = 0.03

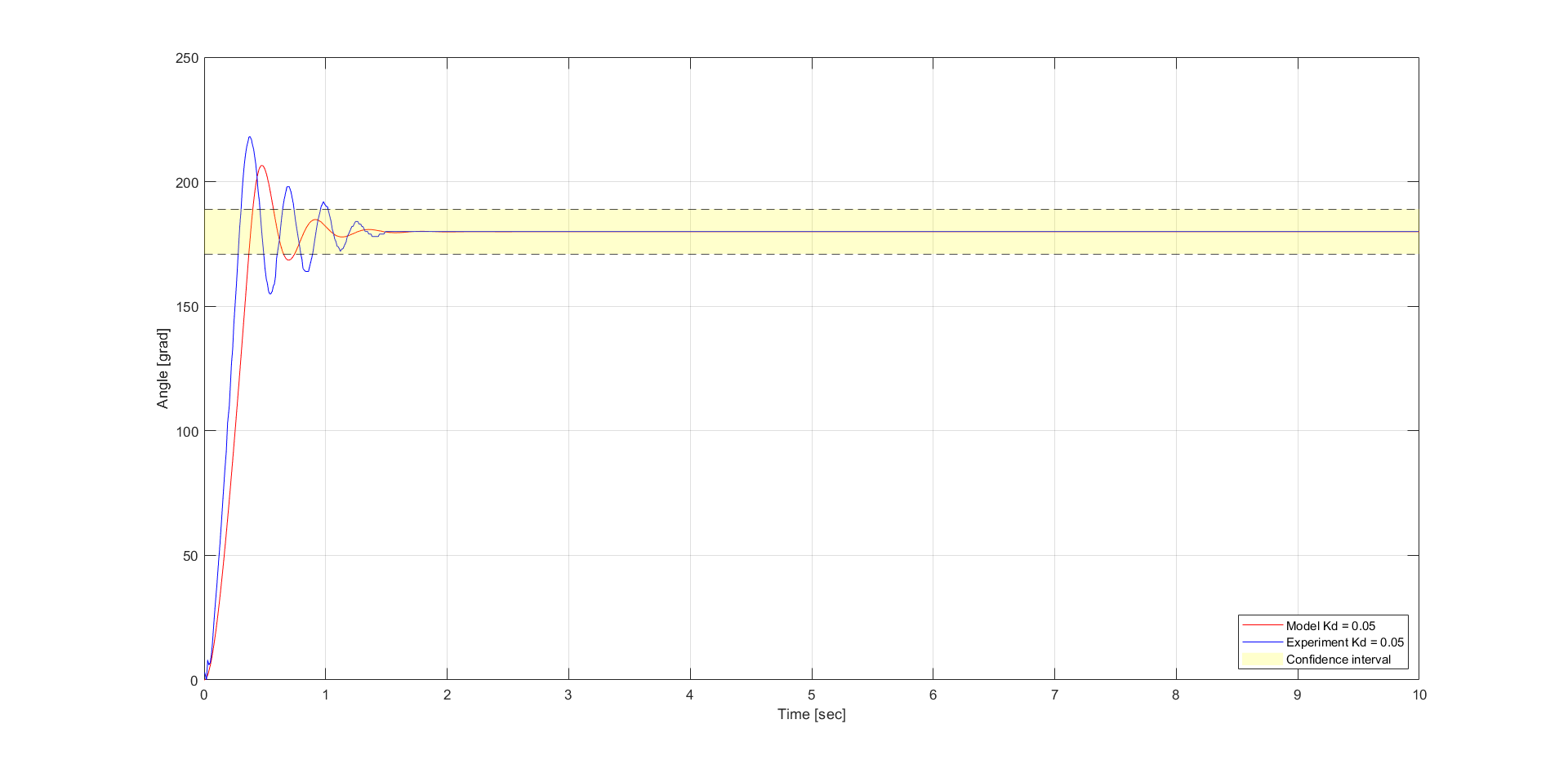


Рисунок 9. Зависимость угла от времени при kd = 0.05

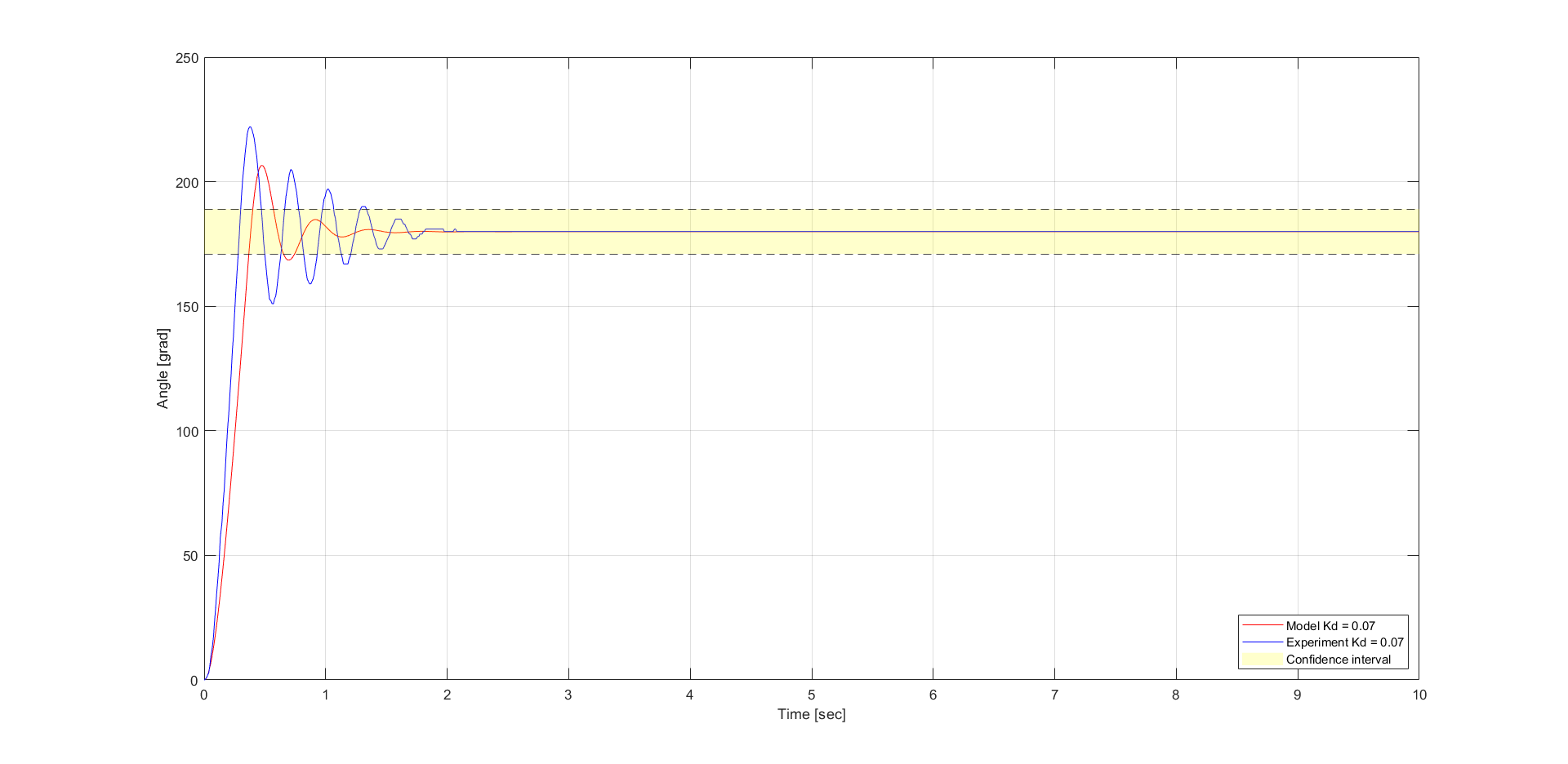


Рисунок 10. Зависимость угла от времени при kd = 0.07

|  |  |  |  |
| --- | --- | --- | --- |
| *Kd* | *e, °* | *σ, %* | *tп,с* |
| 0,005 | 1 | 17,127 | 0,5757 |
| 0,01 | 1 | 17,127 | 0,5723 |
| 0,03 | 0 | 19,4 | 0,7147 |
| 0,05 | 0 | 21,1 | 1,0242 |
| 0,07 | 0 | 23,3 | 1,3489 |
| ↑ | ↓(–) | ↑ | ↑ |

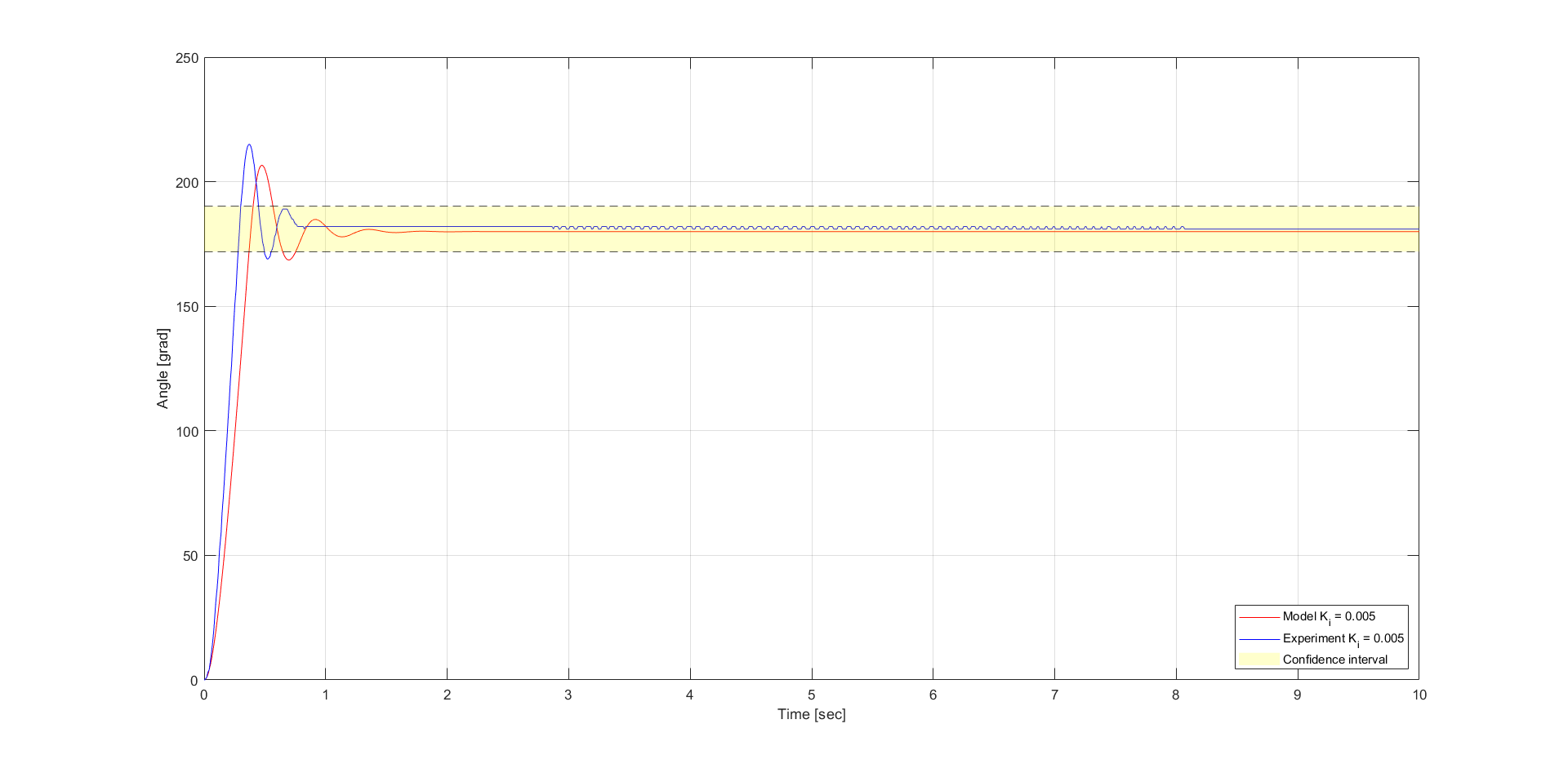
* 1. Изучение ПИ – регулятора  
     Зафиксировав *kp = 4* и *kd = 0,* посмотрим, как будут вести себя *e, σ* и *tп* при изменении *ki*.  
       
     

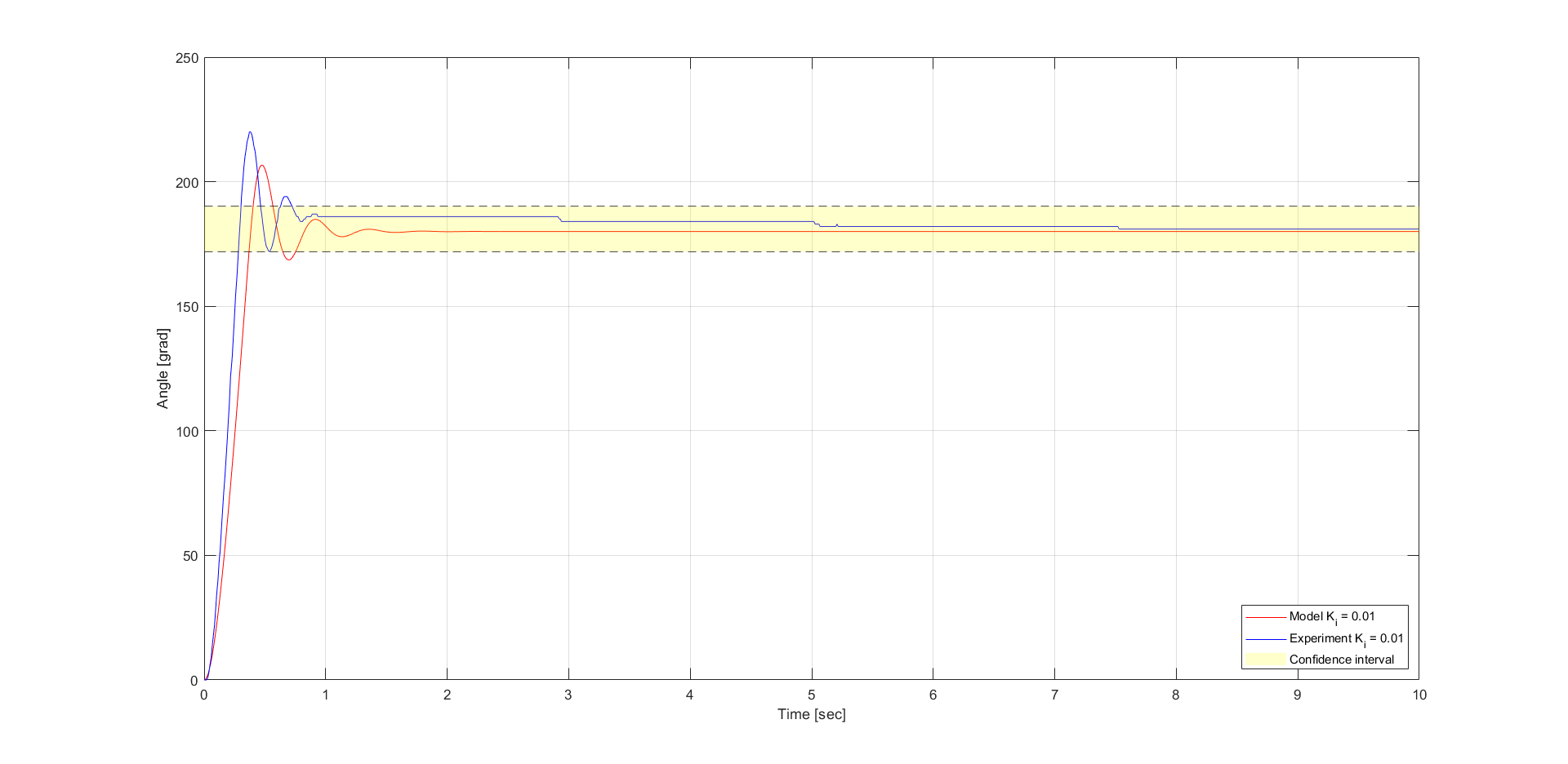
Рисунок 11. Зависимость угла от времени при ki = 0.005  


Рисунок 12. Зависимость угла от времени при ki = 0.01

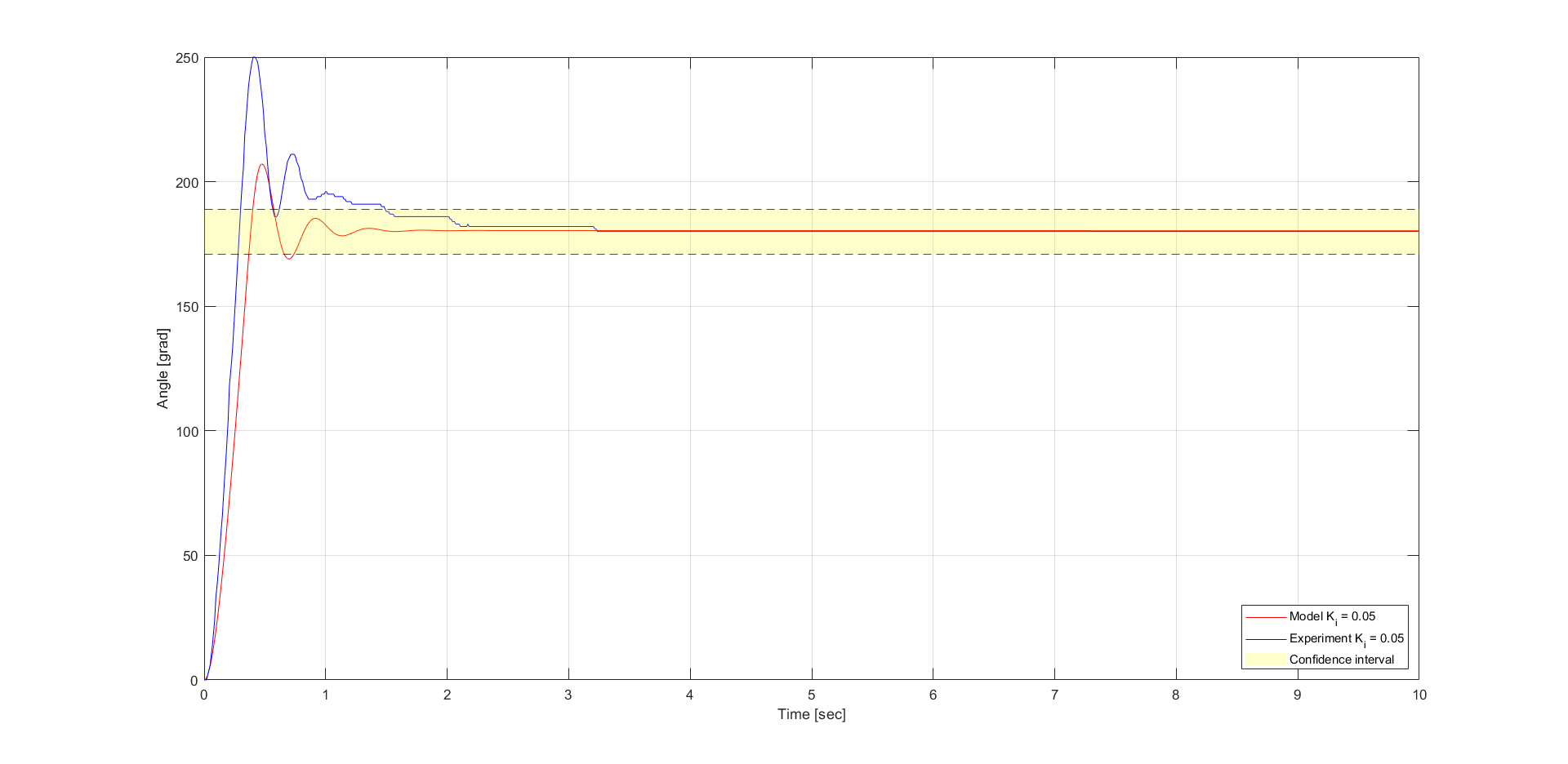


Рисунок 13. Зависимость угла от времени при ki = 0.05

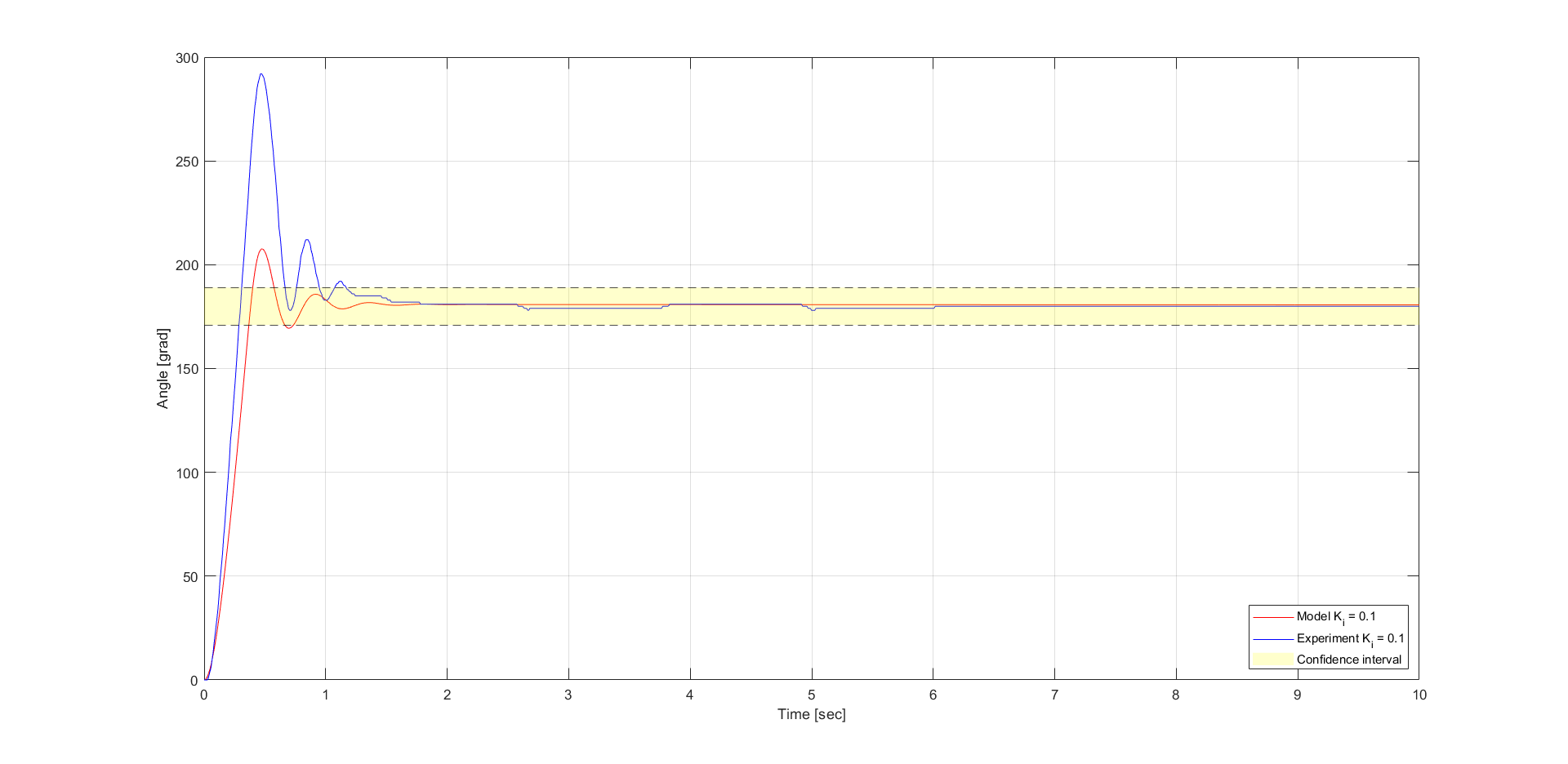


Рисунок 14. Зависимость угла от времени при ki = 0.1

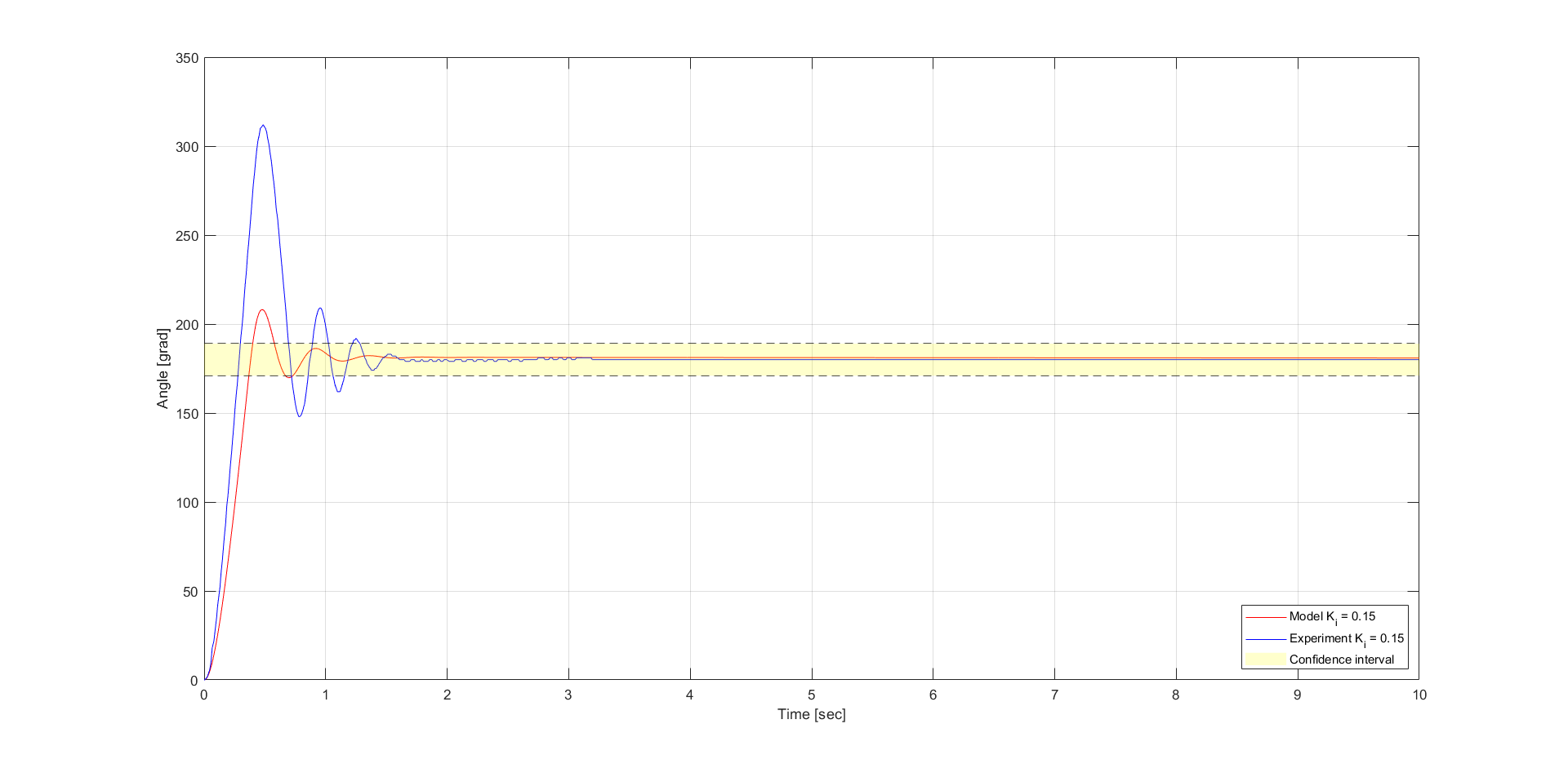


Рисунок 15. Зависимость угла от времени при ki = 0.15

|  |  |  |  |
| --- | --- | --- | --- |
| *Ki* | *e, °* | *σ, %* | *tп,с* |
| 0,005 | 1 | 18,78 | 0,55 |
| 0,01 | 1 | 21,55 | 0,73 |
| 0,05 | 0 | 38,88 | 1,49 |
| 0,1 | 0 | 62,22 | 1,18 |
| 0,15 | 0 | 72,78 | 1,29 |
| ↑ | ↓(–) | ↑ | ↑ |

* 1. Обобщение данных по П, ПИ, ПД – регуляторам  
     Занесем в таблицу изменение *e, σ* и *tп* из предыдущих экспериментов. Красным выделены несоответствующие ожиданиям элементы таблицы

|  |  |  |  |
| --- | --- | --- | --- |
|  | *e* | *σ* | *tп* |
| *Kp* | ↓ | ↑ | ↑ |
| *Ki* | – | ↑ | ↑ |
| *Kd* | – | ↑ | ↑ |

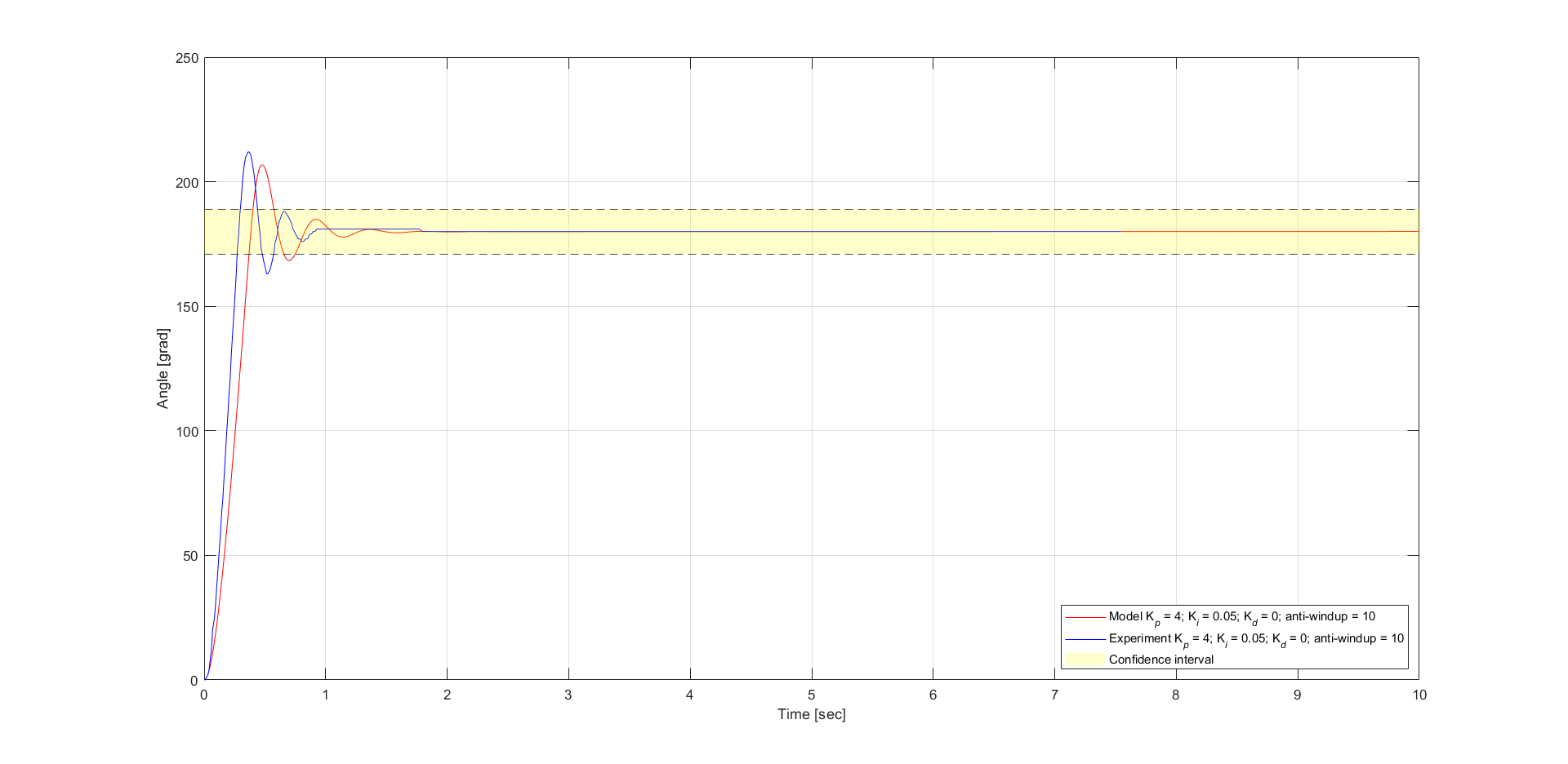
2.5. Изучение ПИ – регулятора с anti – windup

Рисунок 16. Зависимость угла от времени при anti-windup = 10

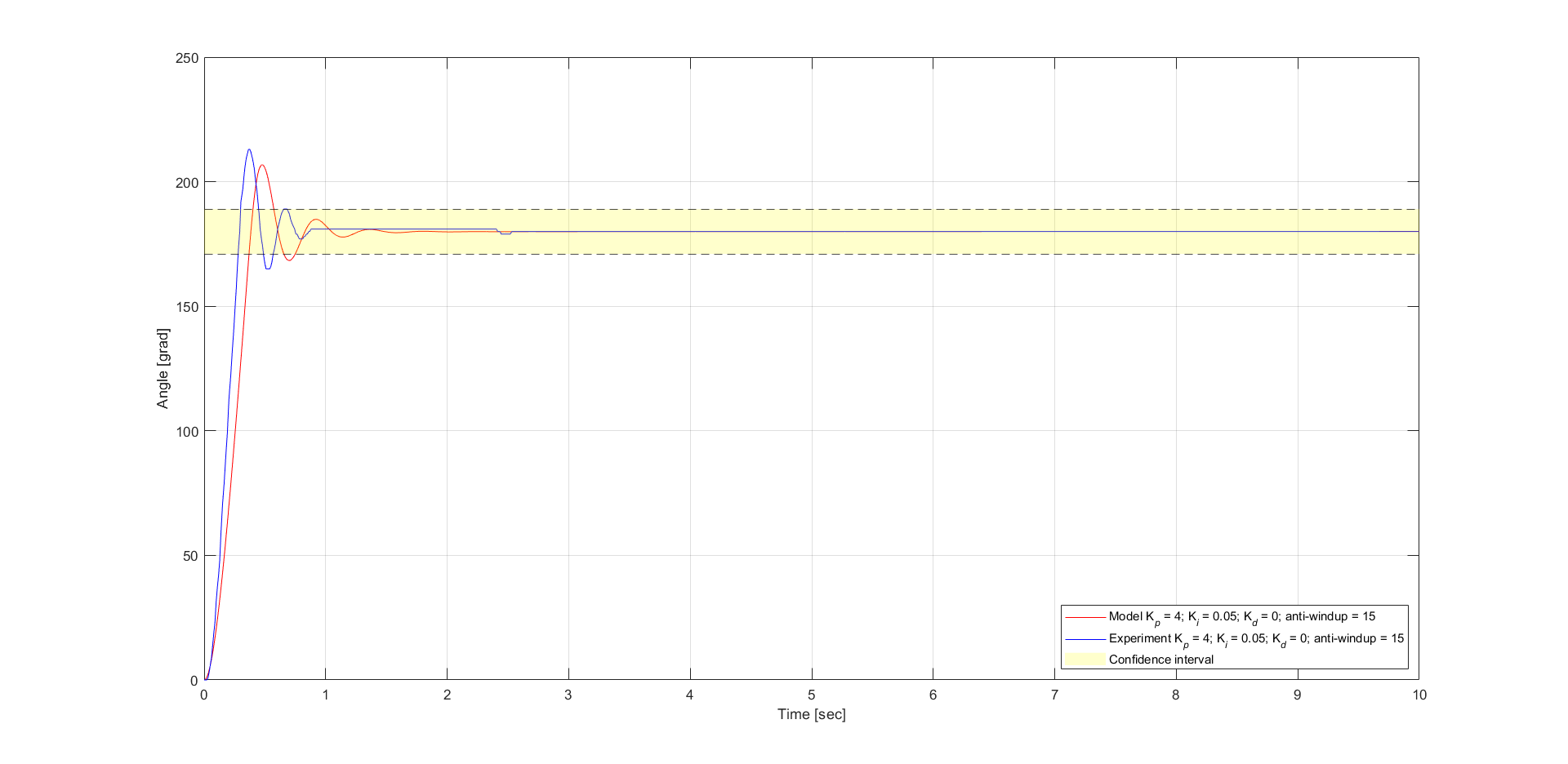


Рисунок 17. Зависимость угла от времени при anti-windup = 15

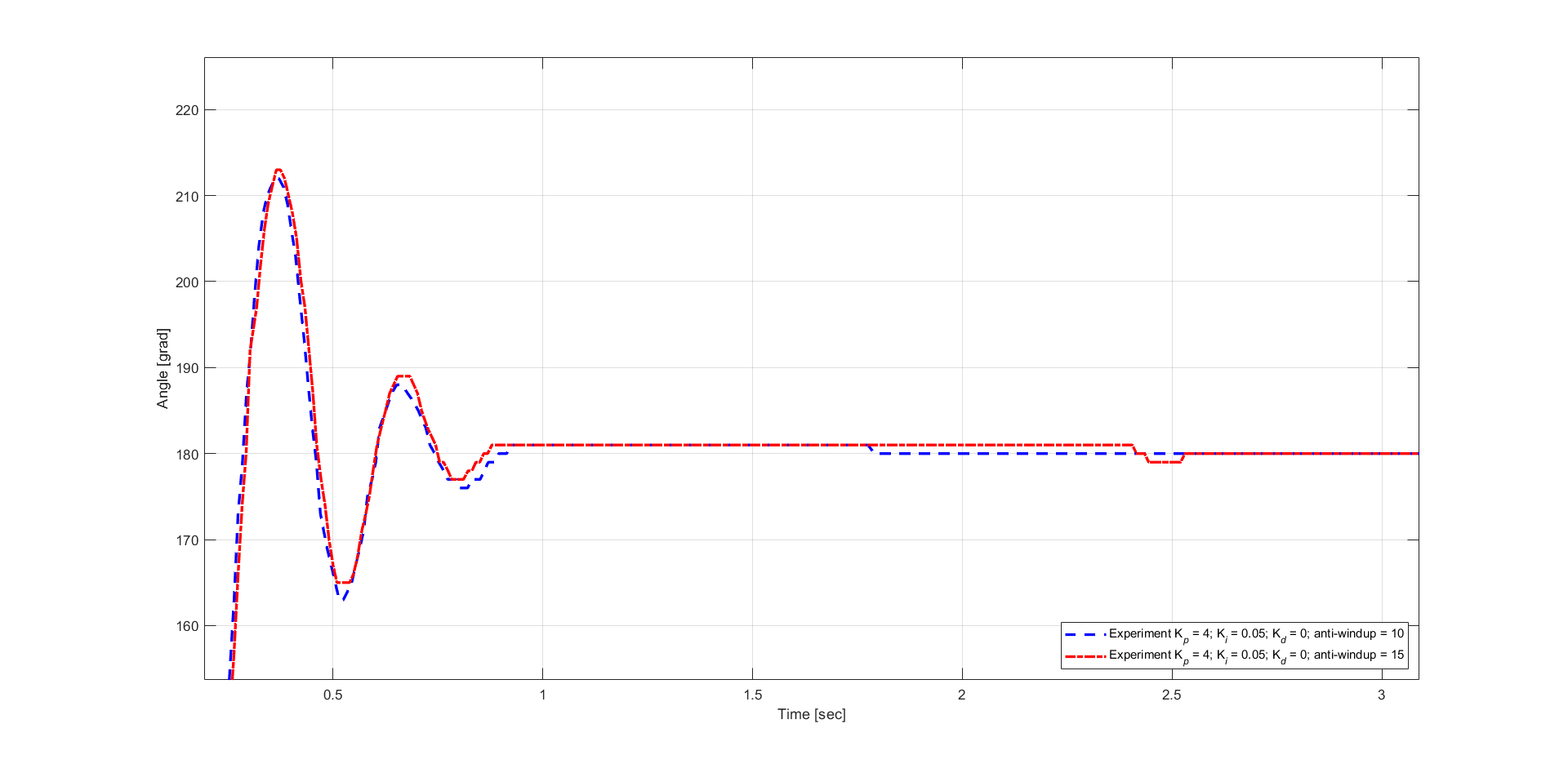


Рисунок 18. Сравнение графиков при anti-windup = 10 и anti-windup = 15

|  |  |  |  |
| --- | --- | --- | --- |
| *anti – windup* | *e, °* | *σ, %* | *tп,с* |
| 10 | 0 | 18,3 | 0,7 |
| 15 | 0 | 17,7 | 0,58 |

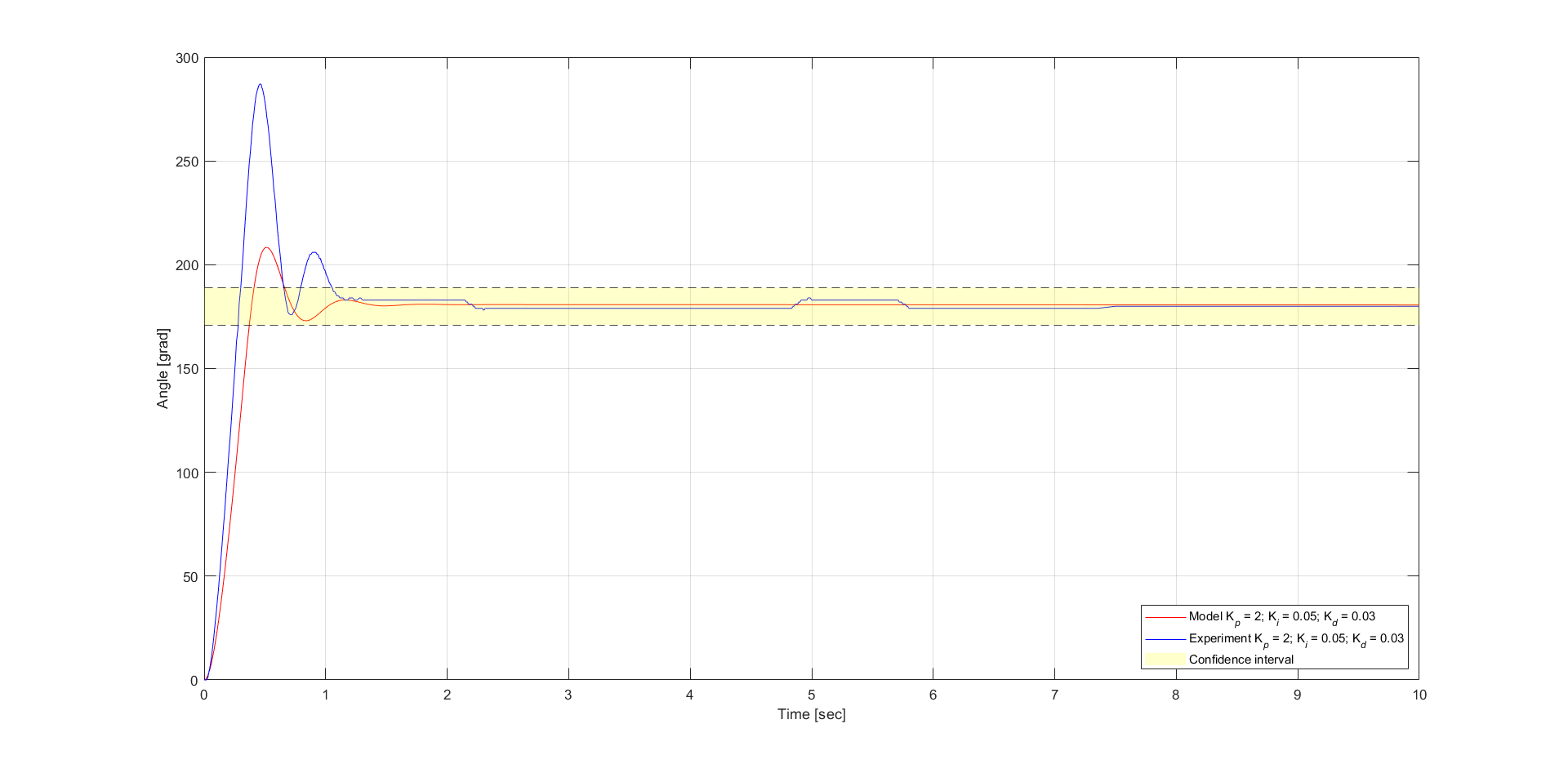
* 1. Изучение ПИД – регулятора  
     

Рисунок 19. Зависимость угла от времени при kp = 2, ki = 0.05, kd = 0.03

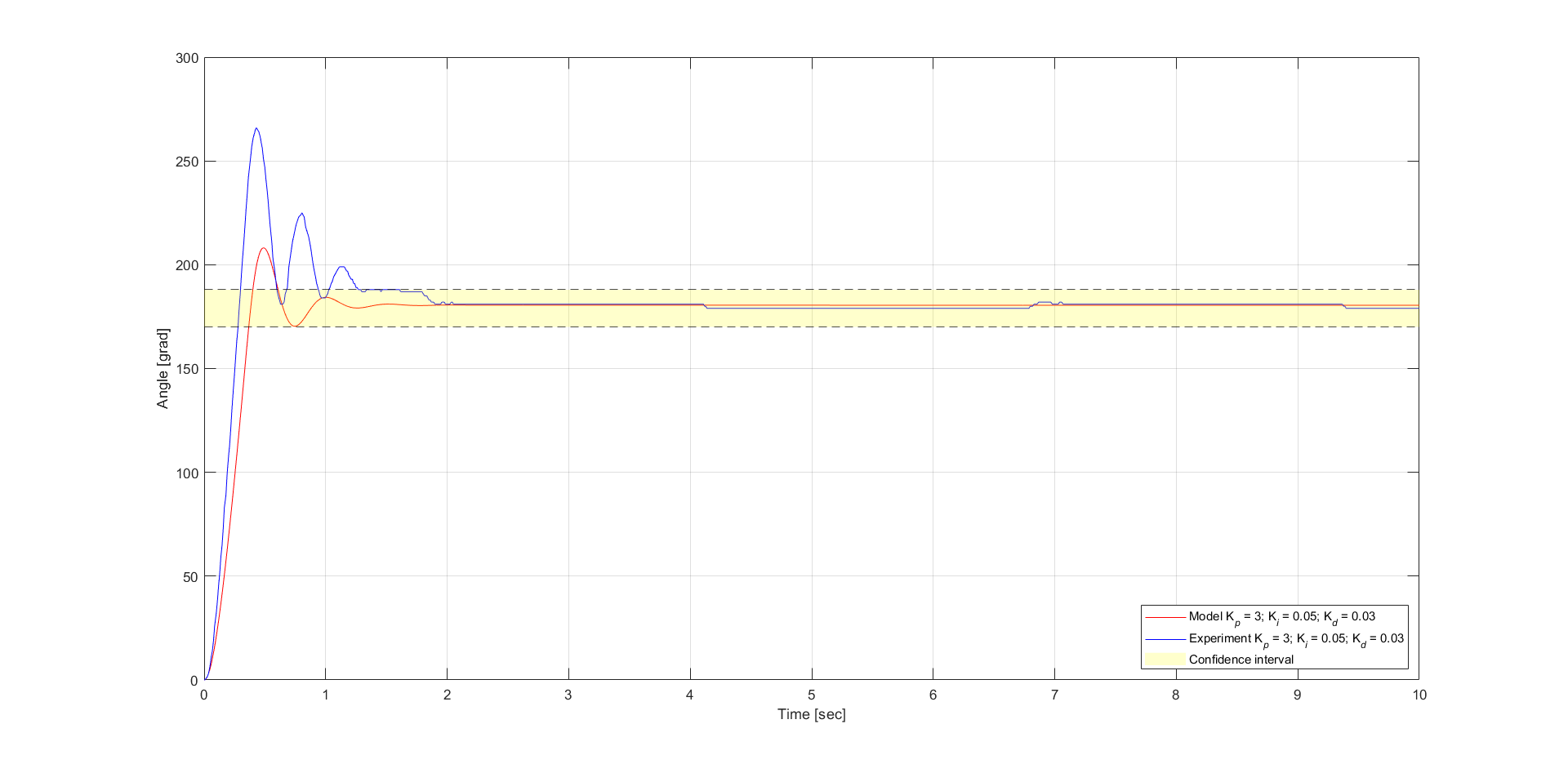


Рисунок 20. Зависимость угла от времени при kp = 3, ki = 0.05, kd = 0.03

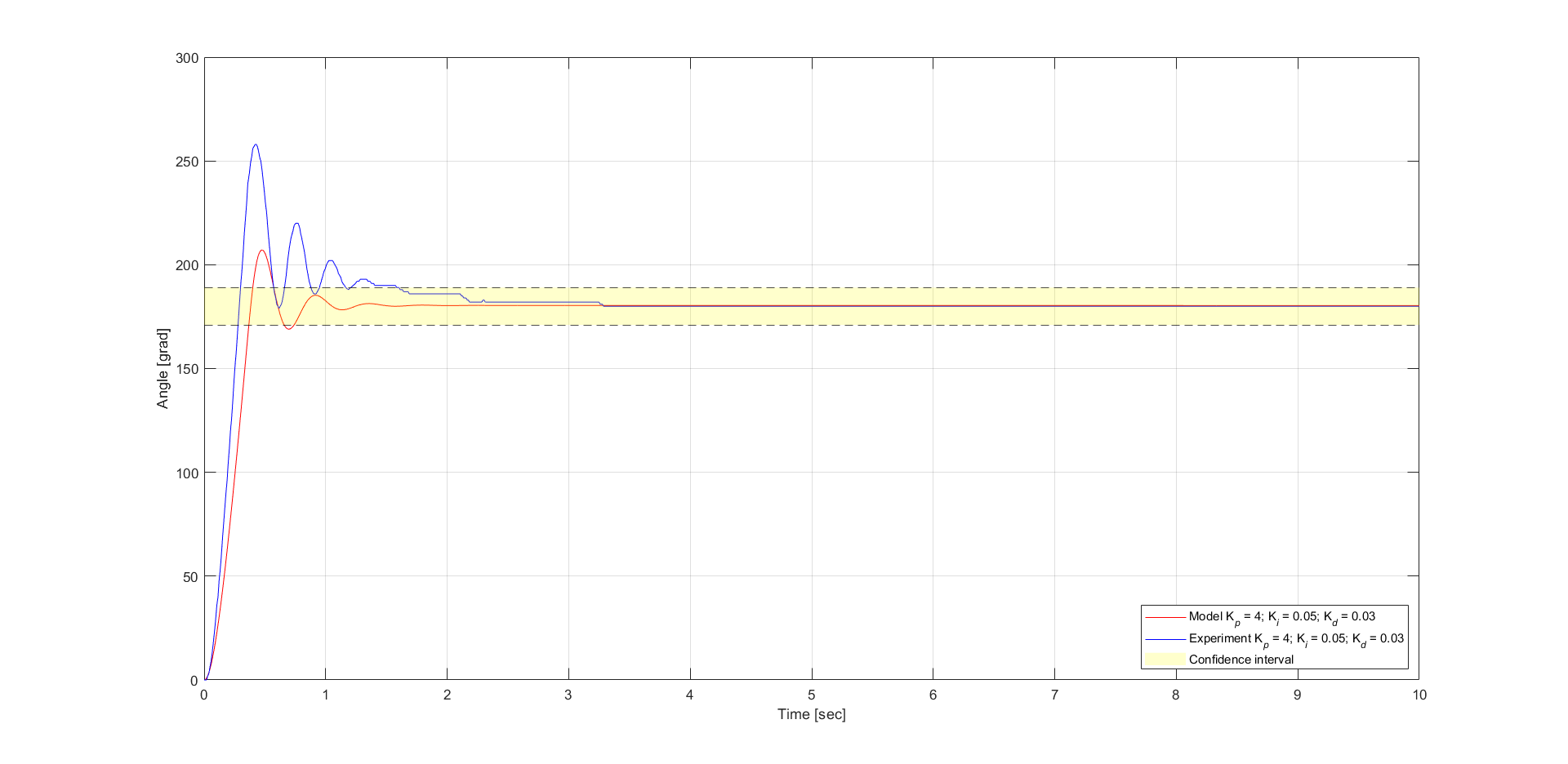


Рисунок 21. Зависимость угла от времени при kp = 4, ki = 0.05, kd = 0.03

|  |  |  |  |
| --- | --- | --- | --- |
| *Kp* | *e, °* | *σ, %* | *tп,с* |
| 2 | 1 | 60,3 | 1,06 |
| 3 | 1 | 48 | 1,6 |
| 4 | 0 | 43 | 1,6 |
| ↑ | ↓ | ↓ | ↑ |

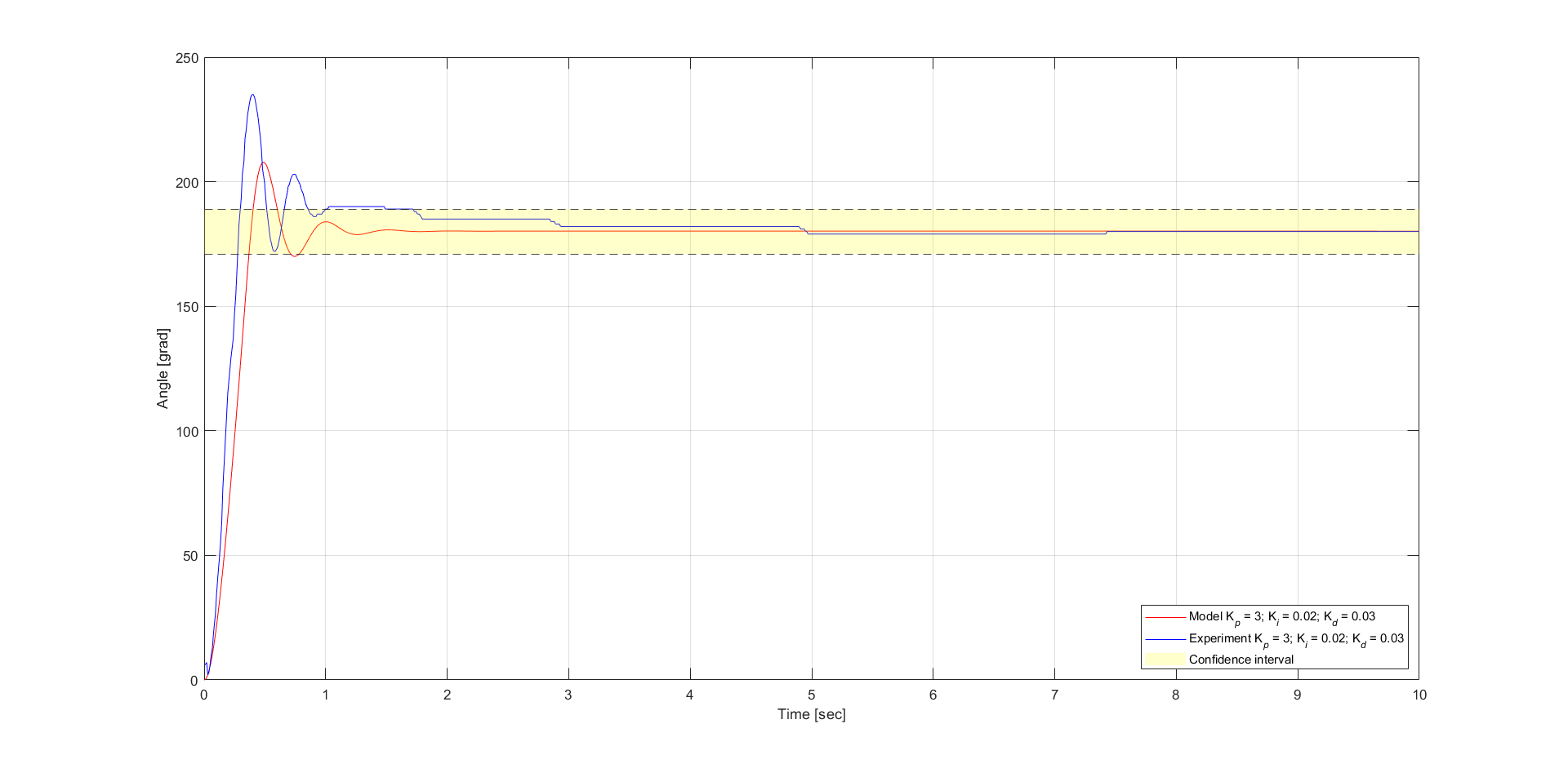


Рисунок 22. Зависимость угла от времени при kp = 3, ki = 0.02, kd = 0.03

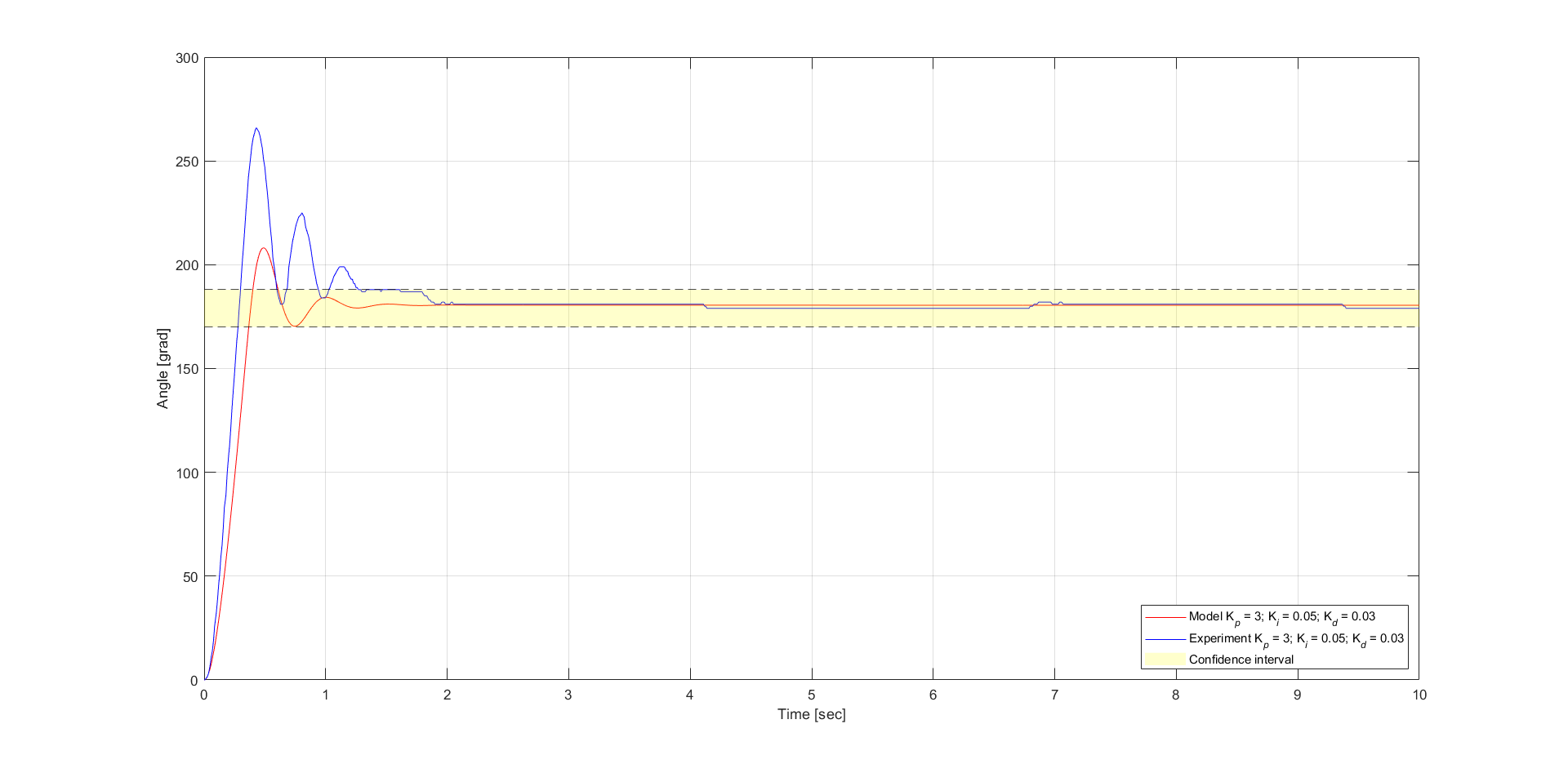


Рисунок 23. Зависимость угла от времени при kp = 3, ki = 0.05, kd = 0.03

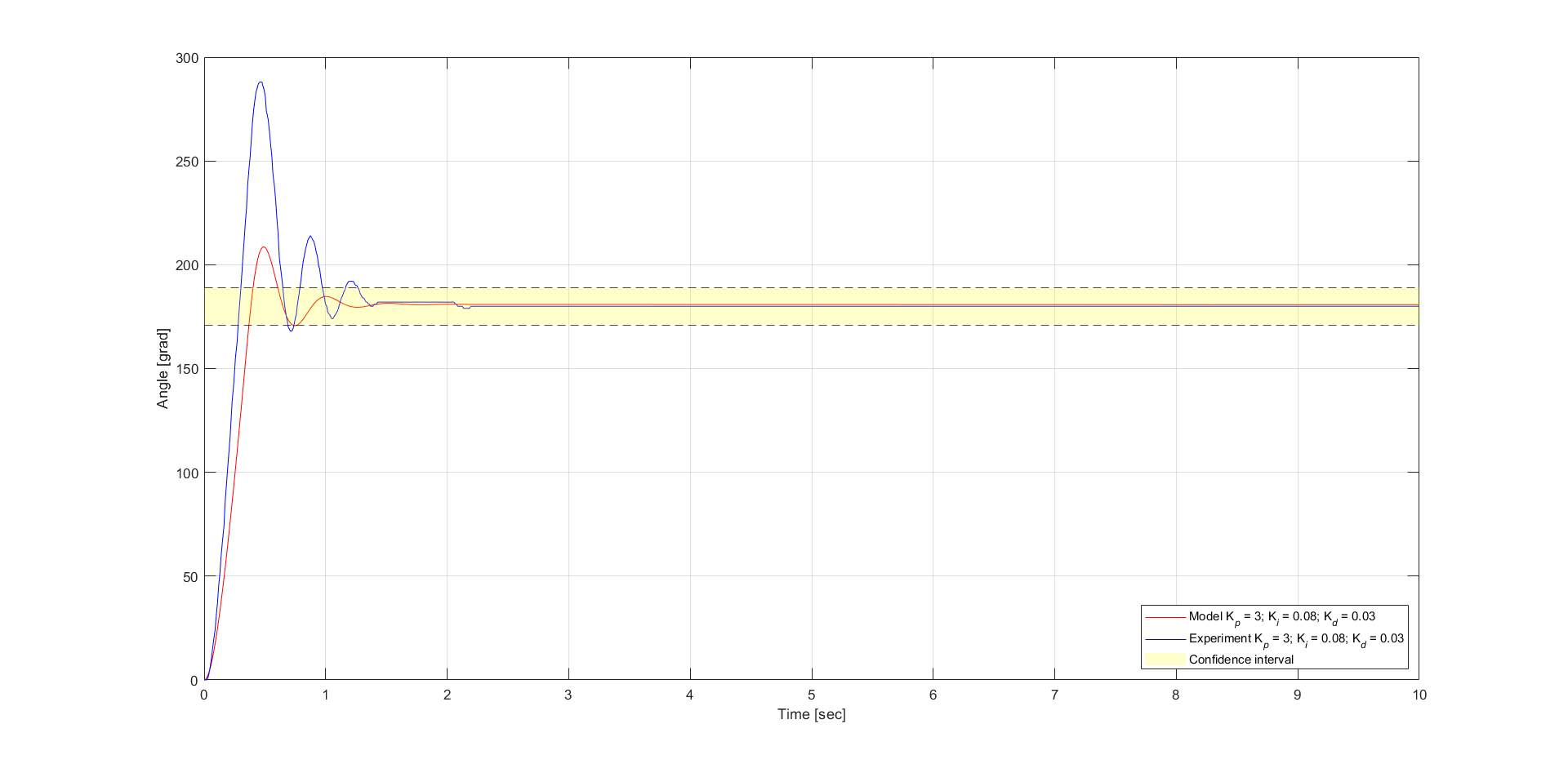


Рисунок 24. Зависимость угла от времени при kp = 3, ki = 0.08, kd = 0.03

|  |  |  |  |
| --- | --- | --- | --- |
| *Ki* | *e, °* | *σ, %* | *tп,с* |
| 0,03 | 1 | 31,3 | 1,75 |
| 0,05 | 1 | 48 | 1,61 |
| 0,08 | 0 | 60 | 1,27 |
| ↑ | ↓ | ↑ | ↓ |

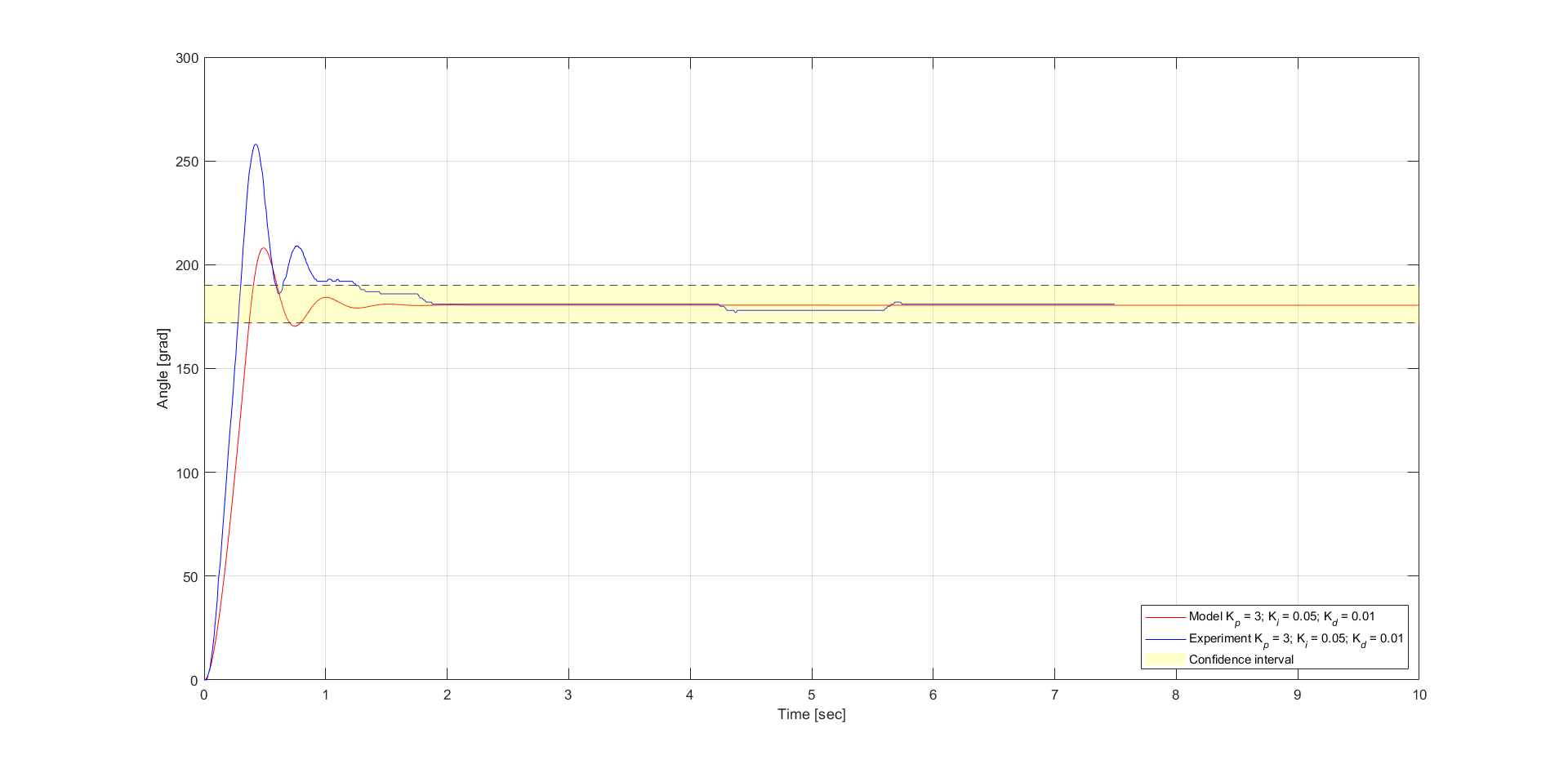


Рисунок 25. Зависимость угла от времени при kp = 3, ki = 0.05, kd = 0.01

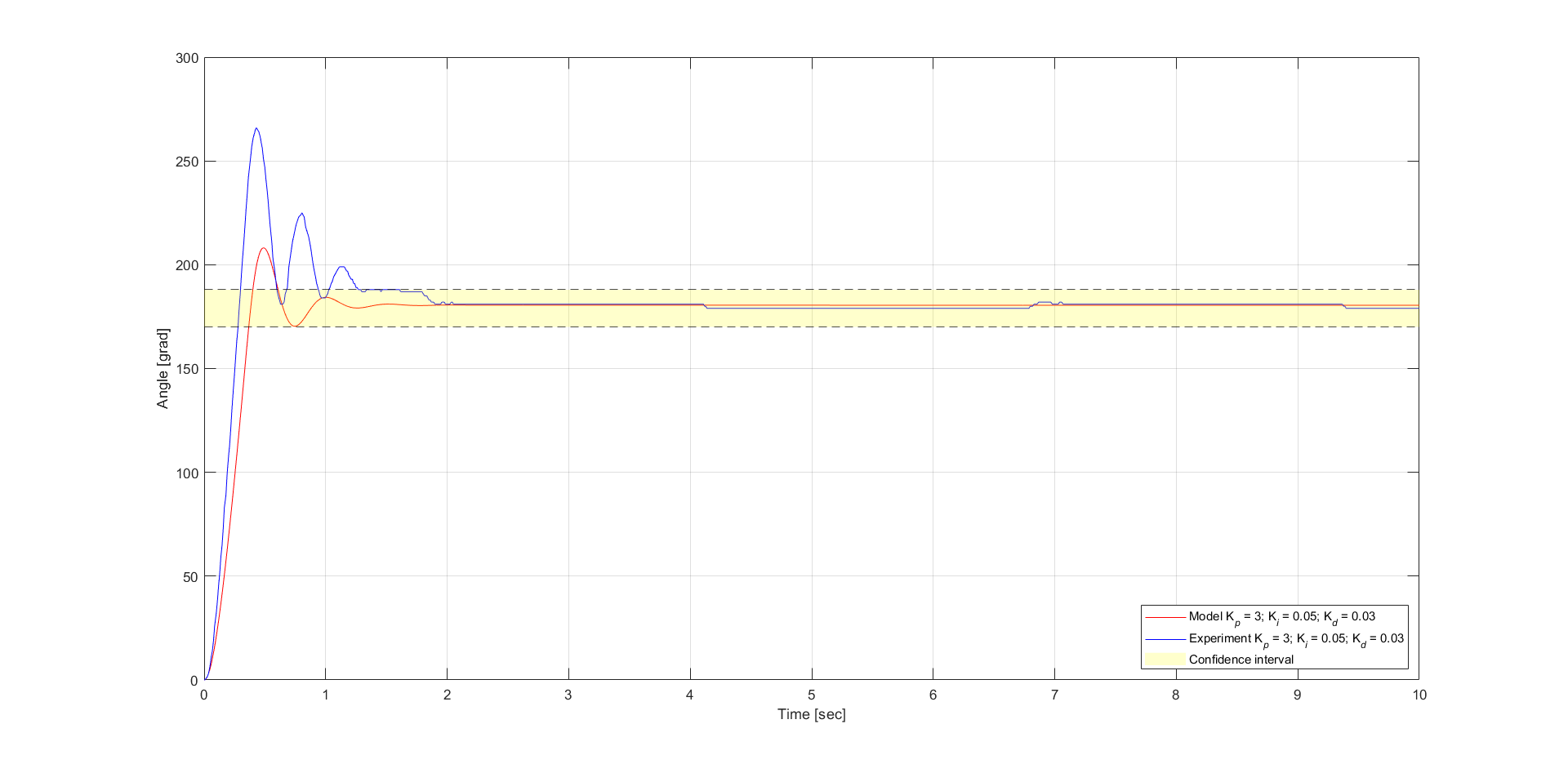


Рисунок 26. Зависимость угла от времени при kp = 3, ki = 0.05, kd = 0.03

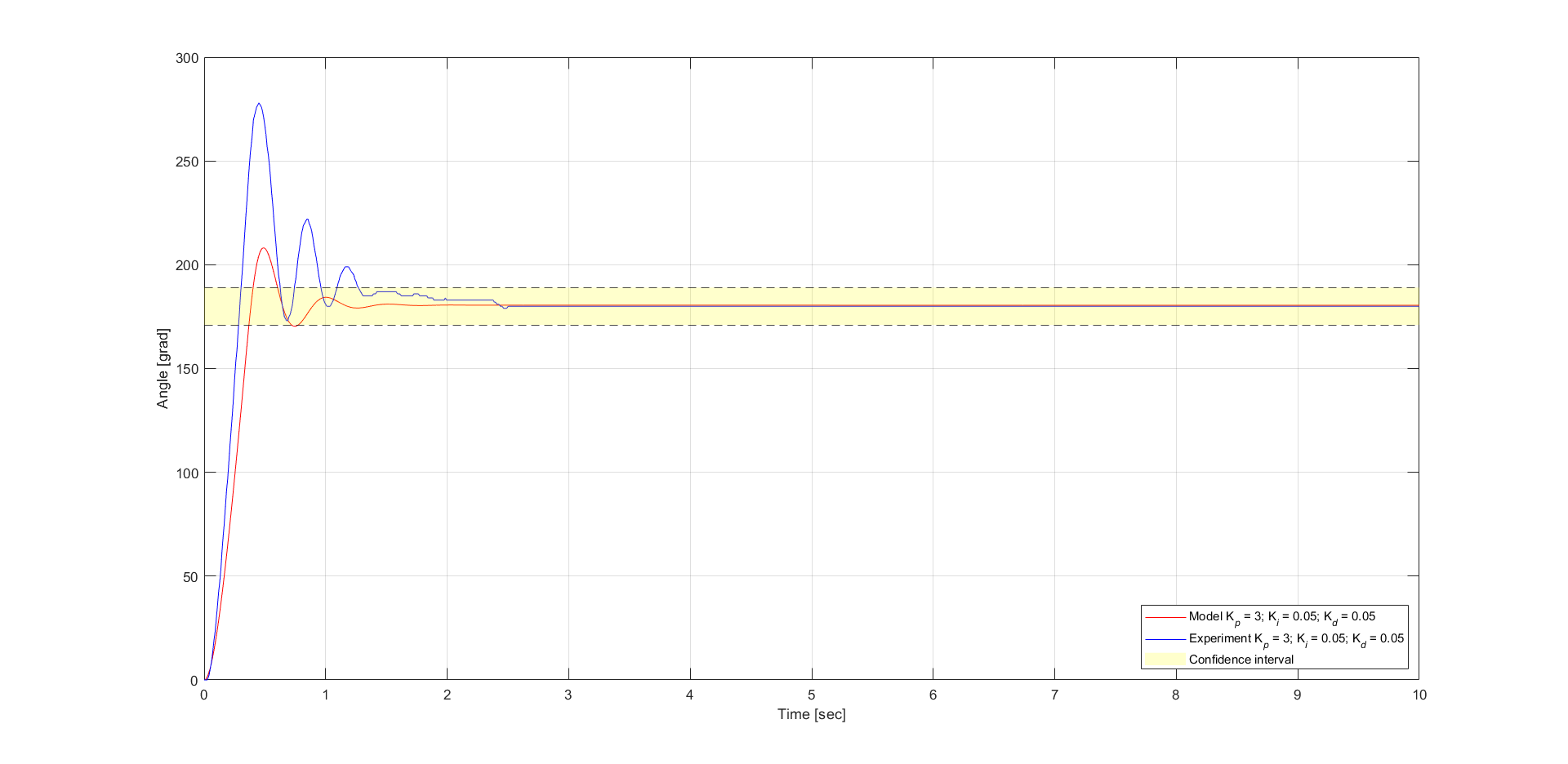


Рисунок 27. Зависимость угла от времени при kp = 3, ki = 0.05, kd = 0.05

|  |  |  |  |
| --- | --- | --- | --- |
| *Kd* | *e, °* | *σ, %* | *tп,с* |
| 0,01 | 1 | 42,5 | 1,25 |
| 0,03 | 1 | 48 | 1,61 |
| 0,05 | 0 | 54 | 1,28 |
| ↑ | ↓ | ↑ | ? |

* 1. Изучение ПИД – регулятора с anti – windup

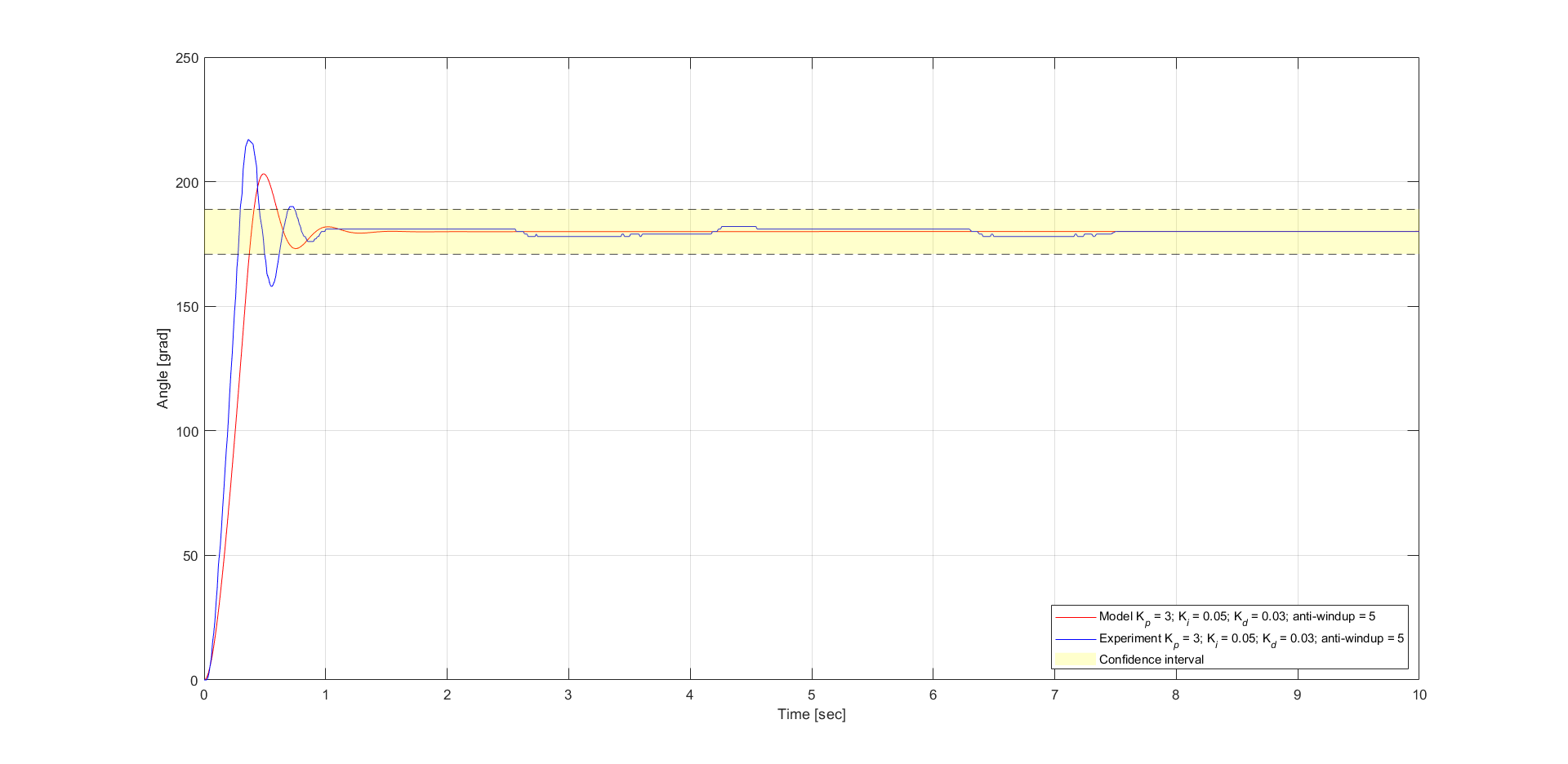


Рисунок 28. Зависимость угла от времени при anti – windup = 5

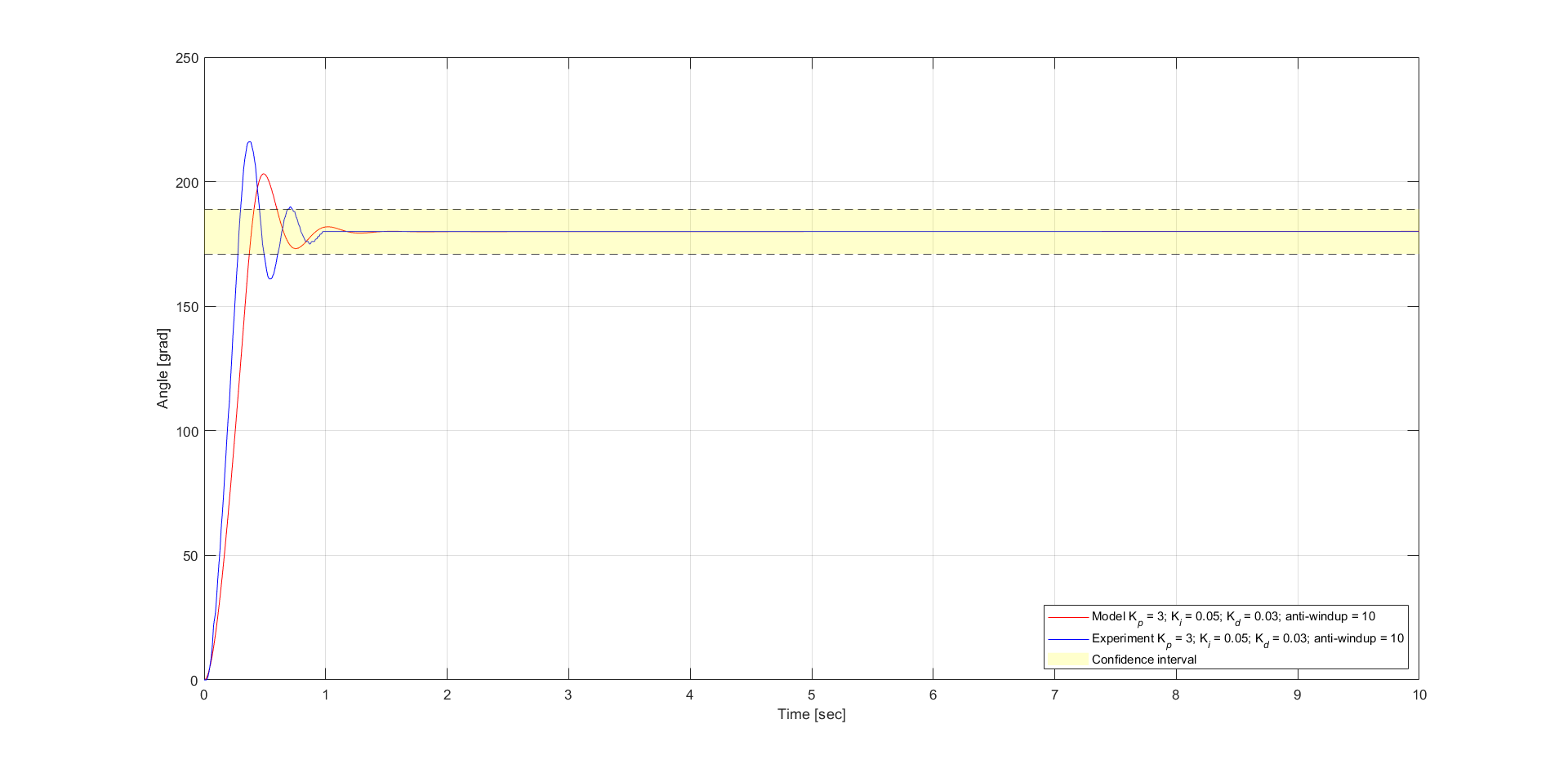


Рисунок 29. Зависимость угла от времени при anti – windup = 10

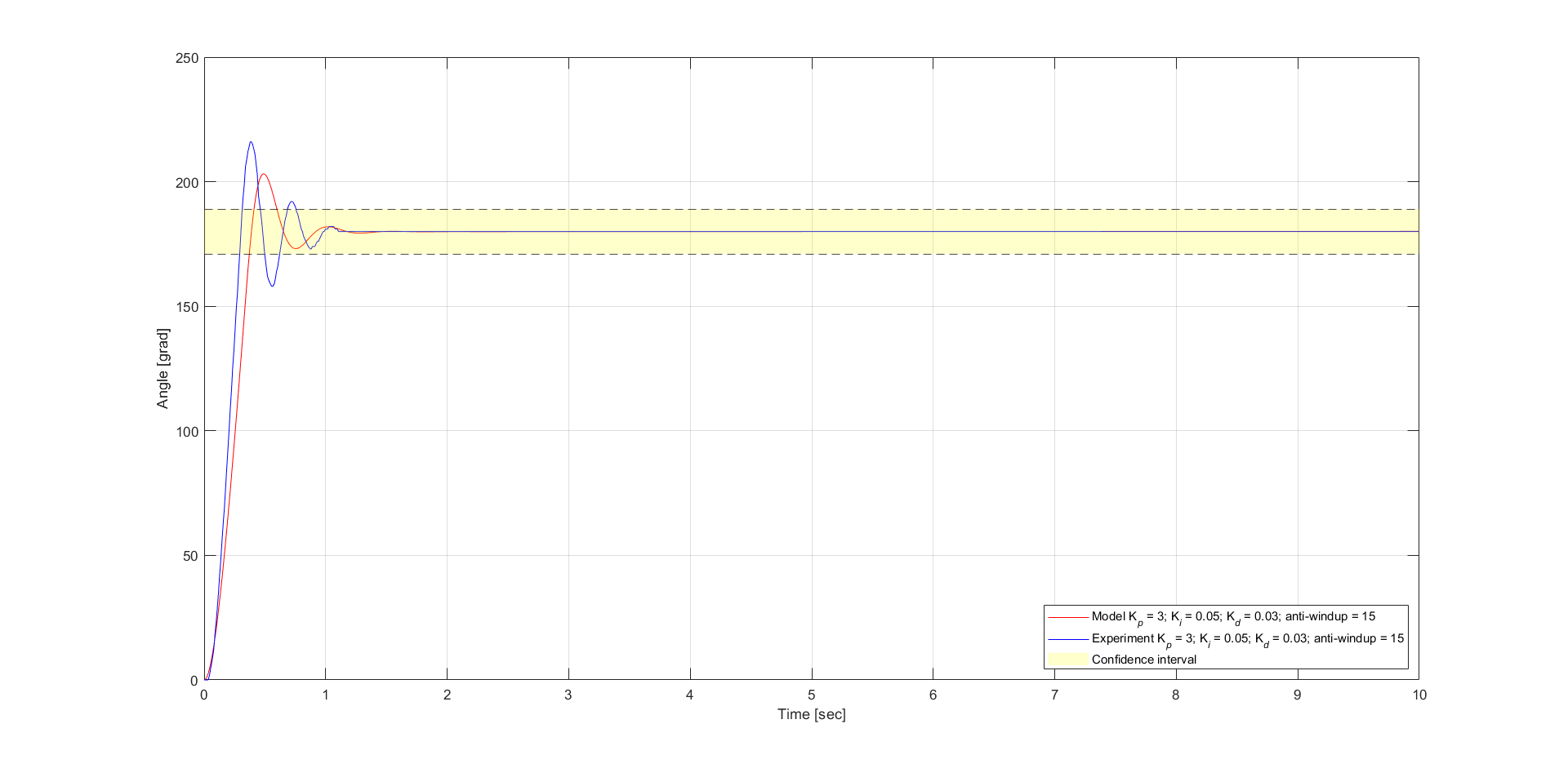


Рисунок 30. Зависимость угла от времени при anti – windup = 15

|  |  |  |  |
| --- | --- | --- | --- |
| *anti – windup* | *e, °* | *σ, %* | *tп,с* |
| 5 | 0 | 20,6 | 0,76 |
| 10 | 0 | 20 | 0,74 |
| 15 | 0 | 20 | 0,76 |

1. Программы для EV3

3.1. ПИД – регулятор для одного двигателя  
  
#!/usr/bin/python3

from ev3dev.ev3 import \*

import time

import math

mC = LargeMotor('C')

Kp = 3

Ki = 0.08

Kd = 0.03

fh = open('data.txt', 'w')

fh.write('0' + ' 0' + '\n')

start\_time = time.time()

mC.position = 0

old\_position = mC.position

old\_time = time.time() - start\_time

sum\_err = 0

while True :

current\_time = time.time() - start\_time

err=180-mC.position

if current\_time < 10:

up = Kp\*err

ud = Kd\*(mC.position - old\_position)/(current\_time-old\_time)

sum\_err = sum\_err + err

ui = Ki\*sum\_err

old\_position = mC.position

old\_time = current\_time

u=up+ui+ud

if abs(u) > 100:

mC.run\_direct(duty\_cycle\_sp = math.copysign(100,u))

fh.write(str(current\_time) + ' ' + str(mC.position) + '\n')

else:

mC.run\_direct(duty\_cycle\_sp = int(u))

fh.write(str(current\_time) + ' ' + str(mC.position) + '\n')

else:

break

mC.stop(stop\_action = 'brake')

time.sleep(1)

fh.close()   
  
3.2. ПИД – регулятор с anti – windup  
  
-//-  
up = Kp\*err

ud = Kd\*(mC.position - old\_position)/(current\_time-old\_time)

if math.fabs(err) > 15:

err\_i = 0

else err\_i = err

sum\_err = sum\_err + err\_i

ui = Ki\*sum\_err

old\_position = mC.position

old\_time = current\_time

u=up+ui+ud  
  
-//-  
  
3.3. ПИД – регулятор для езды вдоль стенки  
  
#!/usr/bin/python3

from ev3dev.ev3 import \*

import time

import math

mB = LargeMotor('B')

mC = LargeMotor('C')

us1 = UltrasonicSensor('in2')

us2 = UltrasonicSensor('in3')

dist1 = us1.value()/10

dist2 = us2.value()/10

Kp = 1.7

Ki = 0.05

Kd = 0.7

aw = 8

h = 17

desired\_dist = 20

fh = open('data.txt', 'w')

fh.write('0' + ' 0' + '\n')

start\_time = time.time()

old\_position = 0.5\*h\*(dist1+dist2)\*math.sqrt(1/(math.pow(h,2)+math.pow((dist1-dist2),2)))

old\_time = time.time() - start\_time

sum\_err = 0

while True :

current\_time = time.time() - start\_time

dist1 = us1.value()/10

dist2 = us2.value()/10

d = 0.5\*h\*(dist1+dist2)\*math.sqrt(1/(math.pow(h,2)+math.pow((dist1-dist2),2)))

err=desired\_dist-d

if current\_time < 20:

up = Kp\*err

ud = Kd\*(d - old\_position)/(current\_time-old\_time)

if math.fabs(err) > aw:

sum\_err = sum\_err + 0

else:

sum\_err = sum\_err + err

ui = Ki\*sum\_err

old\_position = d

old\_time = current\_time

u=up+ui+ud

if dist1 > dist2:

if abs(u) > 30:

mB.run\_direct(duty\_cycle\_sp = 70+math.copysign(30,u))

mC.run\_direct(duty\_cycle\_sp = 70-math.copysign(30,u))

fh.write(str(current\_time) + ' ' + str(d) + '\n')

else:

mB.run\_direct(duty\_cycle\_sp = 70+u)

mC.run\_direct(duty\_cycle\_sp = 70-u)

fh.write(str(current\_time) + ' ' + str(d) + '\n')

else:

if abs(u) > 30:

mB.run\_direct(duty\_cycle\_sp = 70-math.copysign(30,u))

mC.run\_direct(duty\_cycle\_sp = 70+math.copysign(30,u))

fh.write(str(current\_time) + ' ' + str(d) + '\n')

else:

mB.run\_direct(duty\_cycle\_sp = 70-u)

mC.run\_direct(duty\_cycle\_sp = 70+u)

fh.write(str(current\_time) + ' ' + str(d) + '\n')

else:

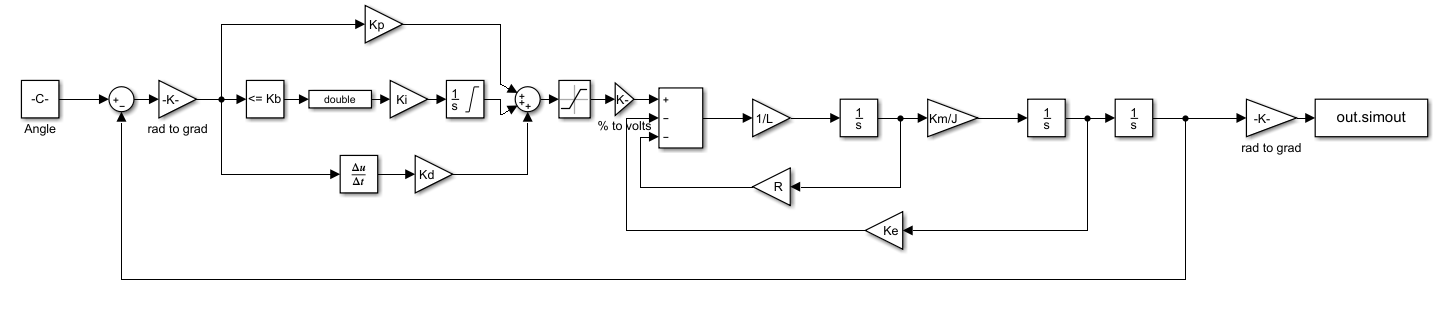
break

mB.stop(stop\_action = 'brake')

mC.stop(stop\_action = 'brake')

time.sleep(1)

fh.close()

1. Модель в Simulink  
   

1. Выводы