controlExperiments

April 1, 2025

1 Introduction

The set of experiments below explore different control functions using PID algorithm. This an incremental approach to the development of the control system.

Here is my nifty citation {cite}`perez2011python`.

1.1 Exp 1 | Simple 1D Rocket Control

In this experiment I am using PID to control a simple model of a rocket in one vertical dimension. The rocket is trying to maintain a constant height, however it is being accelerated downward constantly at a rate of $9.81ms^{-2}$. Its new velocity after a descrete time period, dt is being calculated via the equation :

$v_{n+1} = v_n dt + (\frac{1}{2}a)dt^2$

[20]:
```
import sys
sys.path.append('../')
import os
import time
#from .experiments.pidModule import PidController
#from .experiments.oneDRocket.rocketModel import Rocket
from slap.src.pid.experiments.pidModule import PidController
from slap.src.pid.experiments.oneDRocket.rocketModel import Rocket
from slap.src.pid.experiments.plotter import Visual
import matplotlib.pyplot as plt

sim = Rocket()
visual = Visual()

# --- GAINS ---
KP = 1
KI = 1
KD = 1

target = 0
dt = 1
posPoints = []
controller = PidController(KP, KI, KD)
```

```python
def addXVal(pos):
    posPoints.append(pos)

def plot(posPoints):
    plt.plot(posPoints)
    plt.show()

def Main():

    x = 0
    pos = sim.get_Current()

    while(x < 100):
        power = controller.pid(pos, target, dt)
        sim.update(power, dt)
        pos = sim.get_Current()
        addXVal(pos)
        #visual.visual(pos)
        print(pos,"| Thrust = ", power)
        x = x +1
        #print(x)

        time.sleep(0.01)
    plot(posPoints)

Main()
```

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
Cell In[20], line 13
     10 import matplotlib.pyplot as plt
     12 sim = Rocket()
---> 13 visual = Visual()
     15 # --- GAINS ---
     16 KP = 1

File c:\Users\franc\vscode\projects\slap\slap\src\pid\experiments\plotter.py:11
 ↪in Visual.__init__(self)
      9 MARK_X = 15
     10 MARK_Y = 0
---> 11 marker = turtle.Turtle()
     12 marker.color("red")
     13 marker.speed(1000)
```

```
File C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_3.11.254
↪0_x64__qbz5n2kfra8p0\Lib\turtle.py:3831, in Turtle.__init__(self, shape,
↪undobuffersize, visible)
  3829 if Turtle._screen is None:
  3830     Turtle._screen = Screen()
-> 3831 RawTurtle.__init__(self, Turtle._screen,
  3832                    shape=shape,
  3833                    undobuffersize=undobuffersize,
  3834                    visible=visible)

File C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_3.11.254
↪0_x64__qbz5n2kfra8p0\Lib\turtle.py:2545, in RawTurtle.__init__(self, canvas,
↪shape, undobuffersize, visible)
  2543 TPen.__init__(self)
  2544 screen._turtles.append(self)
-> 2545 self.drawingLineItem = screen._createline()
  2546 self.turtle = _TurtleImage(screen, shape)
  2547 self._poly = None

File C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_3.11.254
↪0_x64__qbz5n2kfra8p0\Lib\turtle.py:527, in TurtleScreenBase._createline(self)
   524 def _createline(self):
   525     """Create an invisible line item on canvas self.cv)
   526     """
--> 527     return self.cv.create_line(0, 0, 0, 0, fill="", width=2,
   528                                capstyle = TK.ROUND)

File <string>:1, in create_line(self, *args, **kw)

File C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_3.11.254
↪0_x64__qbz5n2kfra8p0\Lib\tkinter\__init__.py:2867, in Canvas.create_line(self
↪*args, **kw)
  2865 def create_line(self, *args, **kw):
  2866     """Create line with coordinates x1,y1,…,xn,yn."""
-> 2867     return self._create('line', args, kw)

File C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_3.11.254
↪0_x64__qbz5n2kfra8p0\Lib\tkinter\__init__.py:2849, in Canvas._create(self,
↪itemType, args, kw)
  2847 else:
  2848     cnf = {}
-> 2849 return self.tk.getint(self.tk.call(
  2850     self._w, 'create', itemType,
  2851     *(args + self._options(cnf, kw))))

RuntimeError: main thread is not in main loop
```

1.2 Exp 2 | Simple Boat Model

In this experiment I am using PID to control a simple model of a boat turning in response to the angle of the rudder.

Set out below is the mathmatics behind this simple boat model

(maths)

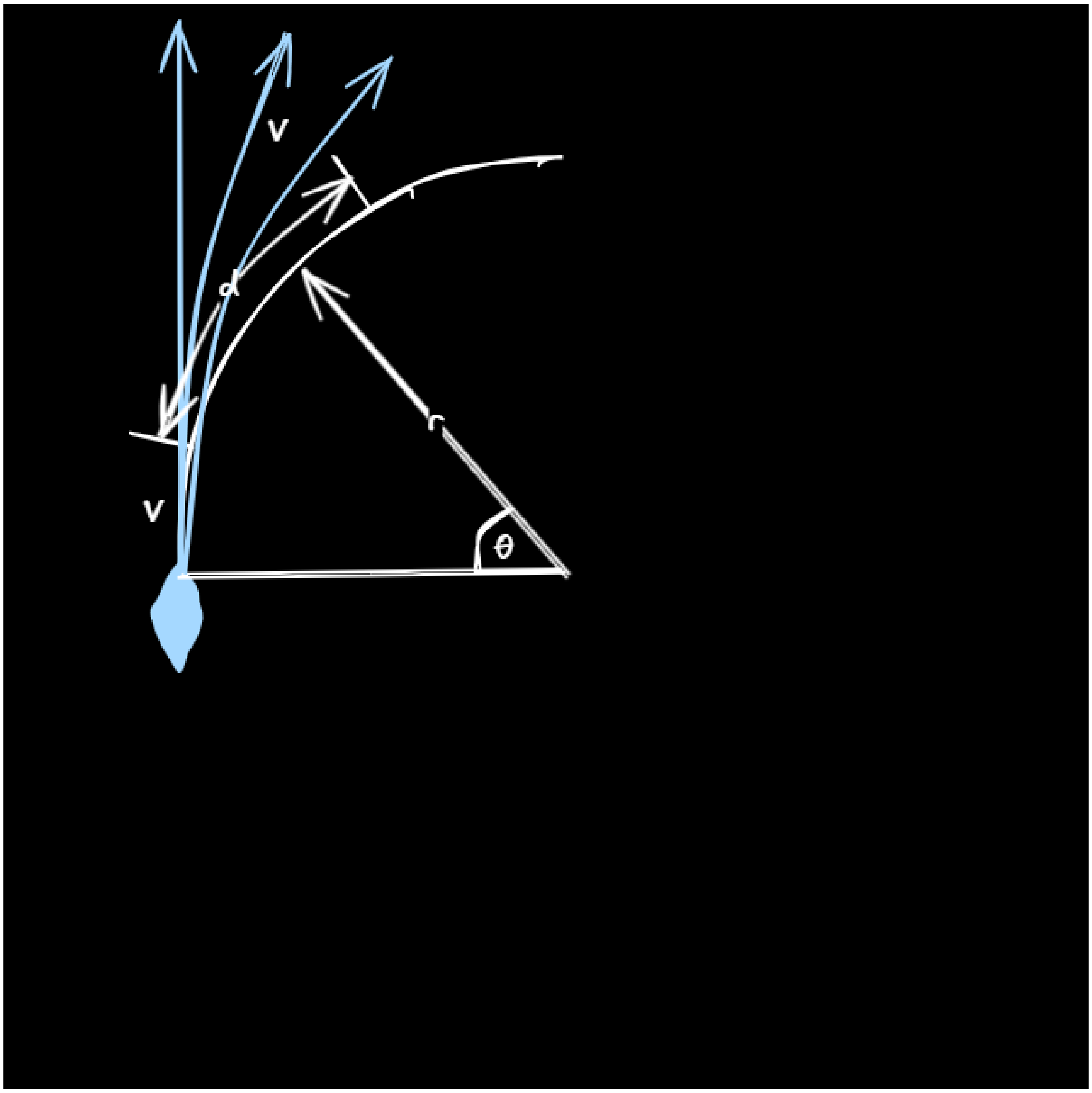
[ ]:
```
import boatv1model
import pid
import tester

set constants: p,i,d
set constant: target heading
using tester with boatv1 model and pid
plot the boat heading over time
```

1.3 Exp 3 | Advanced Boat Model

In this experiment I am using PID to control a advanced model of a boat turning in response to the angle of the rudder, where the boat turn in response to the rudder angle has a time lag.

Set out below is the mathmatics behind this advanced boat model, where the change in the turn is described by a differential equation.

(maths)



[ ]:
```
import boatv2model
import pid
import tester

set constants: p,i,d
set constant: target heading
using tester with boatv2 model and pid
plot the boat heading over time
```

1.4 Estimates of scaling

1.4.1 Rudder Action

To understand the rudders action on the boat, some simple assumptions can be made: - The rudder action is over + or - 25 degrees maximum - When the rudder is fully over (i.e 25 deg), the boat moves over 90 degrees at 5 knots takes 10 seconds

1.5 Direction of Control

We need to understand how to deal with which direction to turn in and the difference between a target heading and an actual heading