Informatique et modélisation

Modélisation et simulation de la commande d'injection d'un moteur à allumage commandé

Nom :

Question 1

Il y a 60 dents donc 120 tdent par tour. Ainsi à la vitesse maximale :

$$tdent = \frac{1}{120} \cdot \frac{1}{\frac{N_{max}}{60}} = \frac{60}{120 \cdot 7000} \simeq 7 \cdot 10^{-5}\ s < 2 \cdot 10^{-3} s = T_e$$

La période d'échantillonnage est trop importante pour détecter le changement d'état.

Question 2

```
def vitesse_moteur(tdent):
    return 60/120/tdent
```

Question 3

$$(nbits)_{10} = 256 \cdot 10^3$$

Question 4

On a 2^{24} adresses donc 16777216 adresses. L'adresse maximale est donc en binaire :

$$(1111\ 1111\ 1111\ 1111\ 1111\ 1111)_2$$

Question 5

$$(320)_{10} = (1\ 0100\ 0000) = (140)_{16}$$

L'avantage de l'hexadécimal est un écriture bien moins lourde.

Question 6

```
def indice(A,val):
    for i in range(1,len(A)):
        if A[i]>val:
            return i-1
```

Question 7

```
def extraire(T,P,Nm,i,j):
    return [[T[i][j],T[i][j+1],P[j],Nm[i]], [T[i+1][j],T[i+1][j+1],P[j+1],Nm[i+1]]]
```

Question 8

```
idp = indice(P,Pcol)
idNm = indice(Nm,Nmot)
ST = extraire(T,P,Nm,idNm,idp)
```

Question 9

```
def interpol(ST,Pcol,Nmot):
    b0 = ST[0][0]
    b1 = ST[0][1]- ST[0][0]
    b2 = ST[1][0]- ST[0][0]
    b3 = ST[1][1]- ST[1][0]- ST[0][1]+ ST[0][0]

    x = (Pcol-ST[0][2])/(ST[1][2]-ST[0][2])
    y = (Nmot-ST[0][3])/(ST[1][3]-ST[0][3])

    return b0+b1*x+b2*y+b3*x*y
```

Question 10

```
def sonde(r):
    lamb = 1/r
    if lamb >=1 :
        return 0.1
    else:
        return 0.9
```

Question 11

```
def duree_injection(Usonde,Kpp,Kpn,Ki,tinjc0,integi,dt):
    if Usonde <=0.5:
        tprop = tinjc0*Kpp
        if integi >=0 :
            integi += Ki*tinjc0*dt
        else:
            integi = 0
    else:
        tprop = tinjc0*Kpn
        if integi <=0:
            integi -= Ki*tinjc0*dt
        else:
            integi = 0

    return (tprop+integi) , integi
```

Question 12

$$s(t_{i+1}) = \frac{K \cdot dt}{\tau} \cdot e(t_i) + \left(1 - \frac{dt}{\tau}\right) \cdot s(t_i)$$

Question 13

```
def Euler(tau,K,dt,si,ei):
    return si+dt*(K*ei-si)/tau
```

Question 14

```
def richesse(tau1,tau2,K,dt,ri,wi,tinji):
    return Euler(tau2,1,dt,ri,wi) , Euler(tau1,K,dt,wi,tinji)
```

Question 15

$$T_{cycle} = \frac{60}{2 \cdot 4000} = 7{,}5 \cdot 10^{-3} s$$

Question 16

```
temps = [t0+k*dt for k in range(int(tn/dt)+1)]
```

Question 17

```
U = liste_U[0]
w = w0
t = 0
for k in range(100):
    for l in range(int(Tcylce/dt)):
        t+=1
        r,w = richesse(tau1,tau2,K,dt,liste_richesse[t-1],w,liste_tinj[t-1])
        liste_richesse.append(r)
        liste_U.append(U)
        tinj,integ = duree_injection(U,Kpp,Kpn,Ki,tinjc0,integ,dt)
        liste_tinj.append(tinj)
    U = sonde(liste_richesse[t])
```

Question 18

Distance la plus courte : 13

Nœud en cours	Liste des distances	Liste des visités
	$[0,+\infty,+\infty,+\infty,+\infty,+\infty]$	[False, False, False, False, False, False]
0	$[0,5,6,+\infty,+\infty,+\infty]$	[True, False, False, False, False, False]
1	$[0,5,6,13{,}8,+\infty]$	[True, True, False, False, False, False]
2	$[0,5,6,13{,}7,15]$	[True, True, True, False, False, False]
4	$[0,5,6,9{,}7,14]$	[True, True, True, False, True, False]
3	$[0,5,6,9{,}7,13]$	[True, True, True, True, True, False]
5	$[0,5,6,9{,}7,13]$	[True, True, True, True, True, True]