```python
import copy
p1=[[1,2,3],[0,3],[0,3,4,0,0]]
p2=[[2,3],[0,2,0]]

#fonctions à 1 variable utiles
def normalize1v (pol1):
    s=copy.deepcopy (pol1)
    while s[-1]==0 and len(s)>1:
        s.pop(len(s)-1)
    return s

def somme1v (pol1,pol2):
    p=copy.deepcopy (pol1)
    q=copy.deepcopy(pol2)
    p1=normalize1v(p)
    q1=normalize1v(q)
    n,m=len(p1),len(q1)
    if n > m :
        s = p1
        for i in range (m) :
            s[i] += q[i]
    else :
        s = q1
        for i in range (n) :
            s[i] += p[i]
    return s

def mult_scal1v (pol,x):
    p=copy.deepcopy (pol)
    s=normalize1v(p)
    for i in range (len(s)):
        s[i] *= x
    return s

def mult_monom1v (pol,x,i):
    """Multiplie le polynôme par X^i"""
    p=copy.deepcopy (pol)
    m=mult_scal1v (p,x)
    s=[0]*i
    return s+m

def mult1v (pol1,pol2):
    p=copy.deepcopy (pol1)
    q=copy.deepcopy (pol2)
    p1,q1=normalize1v(p),normalize1v(q)
    m=[0]
    for i in range (len(p1)):
        s=mult_monom1v(q1,p1[i],i)
        m=somme1v(m,s)
    return normalize1v(m)

#polynome 2 variables
def  normalize2v (pol):
    p=copy.deepcopy(pol)
    n=len(p)
    for i in range (n):
        p[i]=normalize1v(p[i])
    while p != [[0]] and p[-1] == [0]:
        p.pop(-1)
    m=len(p[0])
    n=len(p)
    for k in range (n):
        m=max (m, len(p[k]))
    for i in range (n):
        l=len(p[i])
        while l<m:
            p[i].append(0)
            l=len(p[i])
    return p

def somme2v (pol1,pol2):
    p=copy.deepcopy(pol1)
    q=copy.deepcopy(pol2)
    n=len(p)
    m=len(q)
    if n<m :
        s=copy.deepcopy(q)
        for i in range (n):
            s[i]=somme1v(p[i],q[i])
    else:
        s=copy.deepcopy(p)
        for i in range (m):
            s[i]=somme1v(p[i],q[i])
    return normalize2v(s)

def mult_scal2v (pol,x):
    p=copy.deepcopy (pol)
    s=normalize2v(p)
    for i in range (len(s)):
        for j in range (len(s[0])):
            s[i][j] *= x
    return s

def mult_monom2v (pol,x,i,j):
    p=copy.deepcopy (pol)
    s=mult_scal2v(p,x)
    n=len(s)
    m=len(s[0])
    z1=[0]*j
    for k in range (n):                #rajoute à chaque sous liste autant de zéro
que le degré du monom en Y
        s[k]=z1+s[k]
    z2=[[0]*(m+j)]*i                   #decale le degré du polynome en X
    return z2+s

def mult2v (pol1,pol2):
    p1=copy.deepcopy(pol1)
    p2=copy.deepcopy(pol2)
    s1,s2=normalize2v(p1),normalize2v(p2)
    n,m = len(s1),len(s1[0])
    f=[[0]]
    for i in range (n):
        for j in range (m):
            f=somme2v(f,mult_monom2v(s2, s1[i][j], i, j))
    return f

##Lagrange 2 variables
def lagrangeY (l,i):
    L=[1]
    for j in range (len(l)):
        if j==i:
            L=L
        else:
            L=mult1v(L,[-l[j]/(l[i]-l[j]),1/(l[i]-l[j])])
    return L

def lagrangeX (l,i):
    L=[[1]]
    for j in range (len(l)):
        if j==i:
            L=L
        else:
            L=mult2v(L,[[-l[j]/(l[i]-l[j])],[1/(l[i]-l[j])]])
    return L

def interpole2v (lx,ly,t):
    Lx=[]
    Ly=[]
    n=len(lx)
    m=len(ly)
    inter=[[0]]
    for k in range (n):
        Lx.append(lagrangeX(lx,k)) # Calcul des Lagrangiens selon x
    for l in range (m):
        Ly.append([lagrangeY(ly,l)]) # Calcul des Lagrangiens selon y
    for i in range (n):
        for j in range (m):
            inter=somme2v(inter,mult_scal2v(mult2v(Lx[i],Ly[j]),t[i][j])) # Somme pour
chaque point
    return inter
```