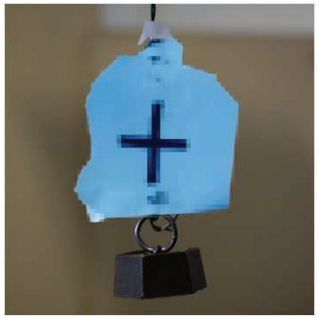
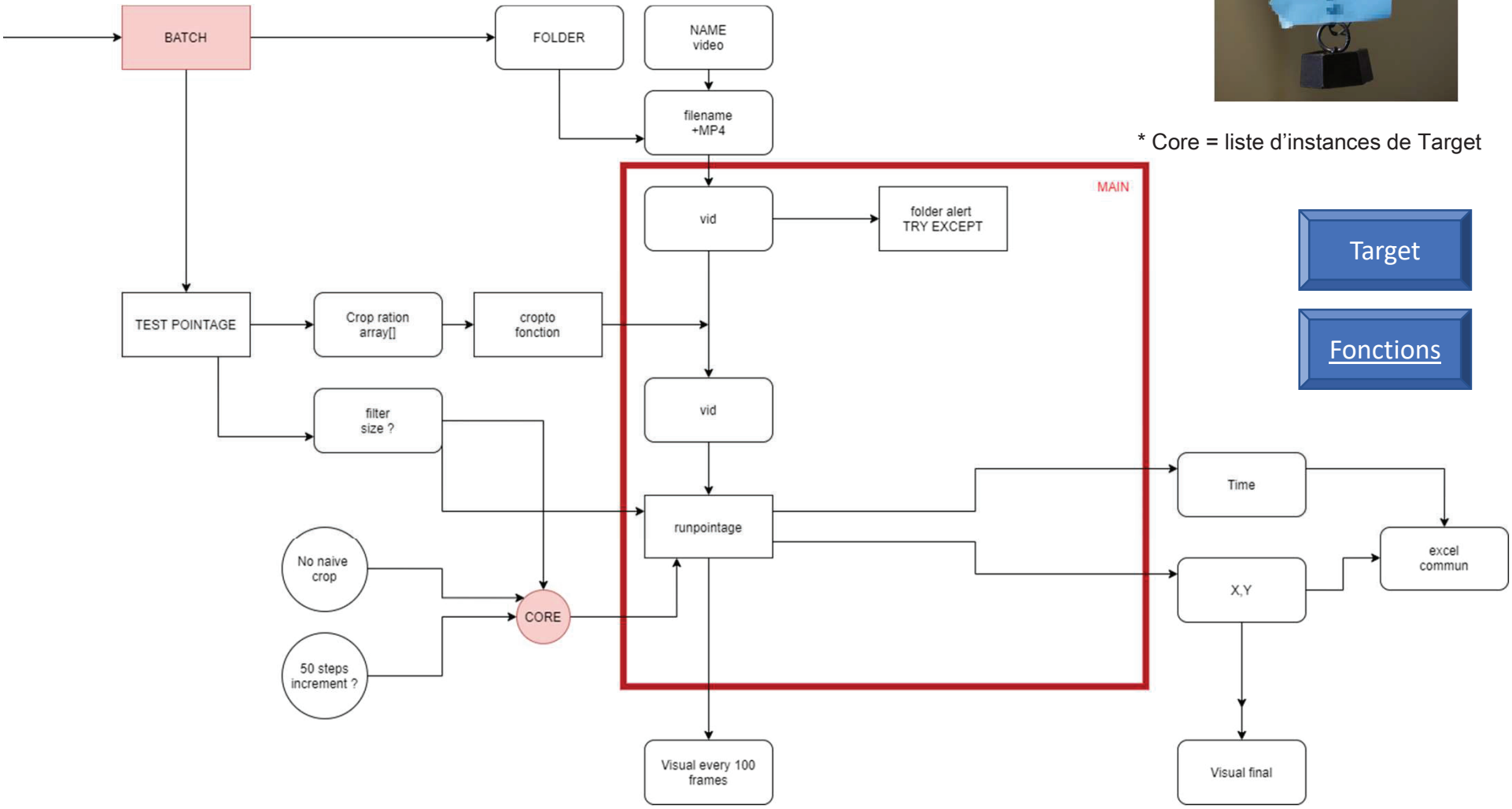
Une solution algorithmique

Obstacle	Solution
Contenir l'information de pointage entre les différentes vidéo et entre objets à pointer	Target class
Complexité temporelle	Pointage suivi paramètre STEP
Images RGB	Contraste en tons de gris
Aisance d'utilisation	Interface utilisateur avec la console

Une solution algorithmique



* Core = liste d'instances de Target



Réalisé avec draw.io

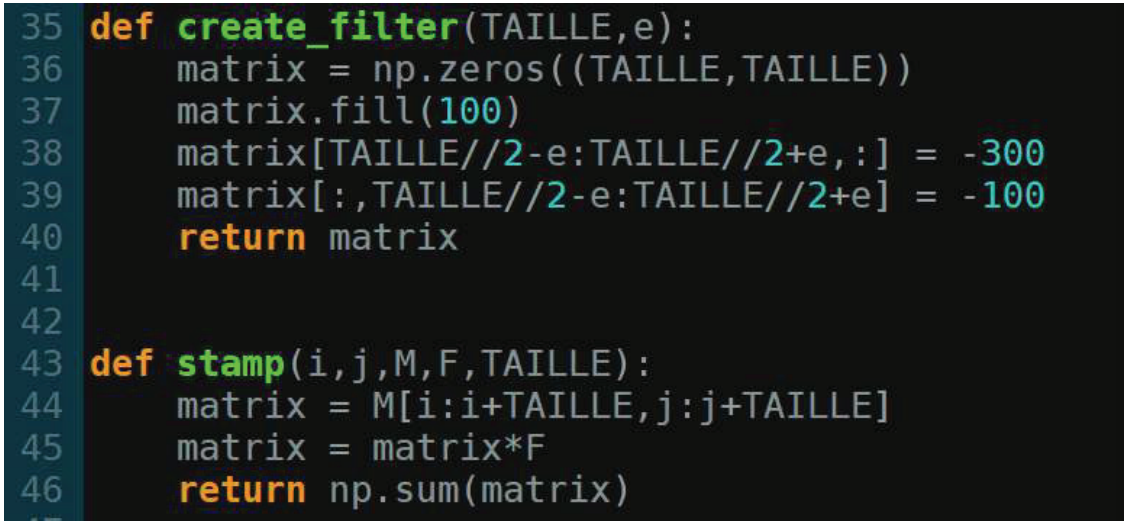
Python: Target class

TAILLE, LINE : paramètres du filtre
STEP : paramètre de pointage dynamique

```python
class Target():
    # str to show is 'X' or 'Y'
    def __init__(self,NAME,x1,x2,y1,y2,TAILLE,STEP,LINE,str_to_show):
        self.NAME = NAME
        self.x1 = x1
        self.x2 = x2
        self.y1 = y1
        self.y2 = y2
        self.TAILLE = TAILLE
        self.STEP = STEP
        self.LINE = LINE
        self.str_to_show = str_to_show
        self.num = True
        self.X = []
        self.Y = []
        self.i = 0
        self.j = 0

    def __str__(self):
        return 'my name is '+NAME

    def process_img(self,image):
            image = image[self.x1:self.x2, self.y1:self.y2]
            image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)

            if self.num==True:
                self.i,self.j = find_pos(image,self.STEP,self.TAILLE,self.LINE,return_type='pos')
                self.num = False
            else:
                self.i,self.j = find_pos(image,self.STEP,self.TAILLE,
self.LINE,return_type='pos',old_i=self.i,old_j=self.j)
            self.X.append(self.i)
            self.Y.append(self.j)
            return self.i, self.j

    def get_XY(self):
        return self.X, self.Y

    def reset(self):
        self.num = True
        self.X = []
        self.Y = []
```

Python: fonctions create_filter & stamp

Utilisation des matrices Numpy (Arrays).



```
35 def create_filter(TAILLE,e):
36     matrix = np.zeros((TAILLE,TAILLE))
37     matrix.fill(100)
38     matrix[TAILLE//2-e:TAILLE//2+e,:] = -300
39     matrix[:,TAILLE//2-e:TAILLE//2+e] = -100
40     return matrix
41
42
43 def stamp(i,j,M,F,TAILLE):
44     matrix = M[i:i+TAILLE,j:j+TAILLE]
45     matrix = matrix*F
46     return np.sum(matrix)
```

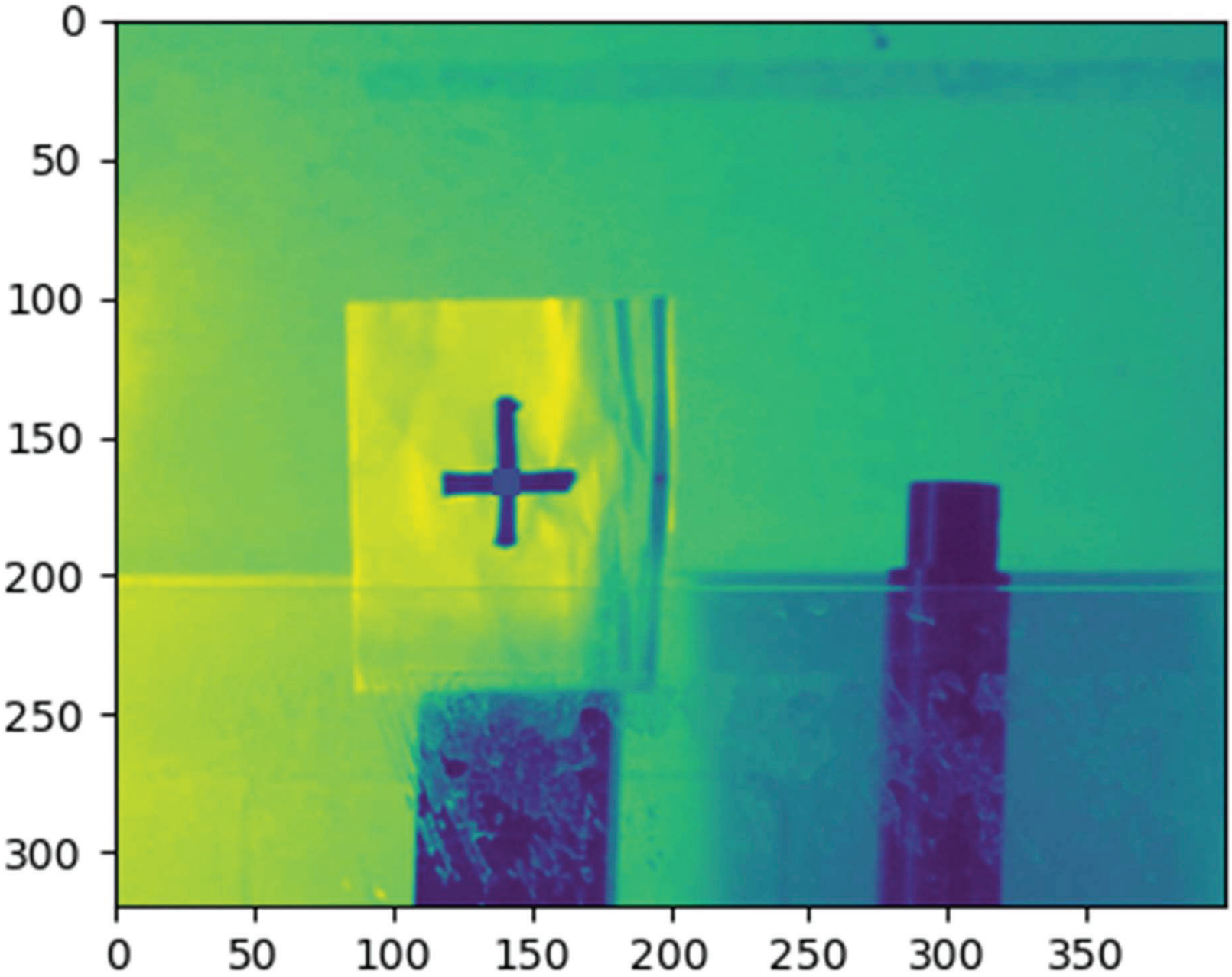


Photo modifiée avec pointage

Une solution algorithmique

Toujours plus vite…

Performance moyenne de 0,16 secondes par image.
Performance moyenne de 0,07 secondes avec la recherche rapide !

Essai avec carte graphique (GPU procession) : 3x performance

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Image caractéristiques : 720 * 1280.												
2	x1						tpf = time per frame						
3	0			CPU	0,36911721			no GPU		GPU		GPU cropped	
4	x2			GPU auto typed	0,15558605			frames count	time	f count	time	f count	time
5	500			GPU cropped	0,06495549			444	163,86	444	70,94	444	29,9
6	y1							512	188,31	512	78,89	512	33,09
7	300			ratio CPU/GPU	2,37243122			302	113,65	302	46,65	302	19,48
8	y2			ration crop/uncropped	2,39527181			536	198,39	536	83,06	536	34,73
9	1200							410	150,34	410	63,8	410	26,41
10	TAILLE							492	180,59	492	76,12	492	31,51
11	50												
12	LINE												
13	5							..		..		..	
14	STEP												
15	80												
16	parametres un peu extremes												
17													
18													
19													
20													
21													