PAr 126

Visualisation des équilibres
de réseau de transport

Clément BORNE - Fabien DURANSON

Introduction - Contexte

- Visualisation = interface homme-machine :
  - comprendre
  - pas universelle

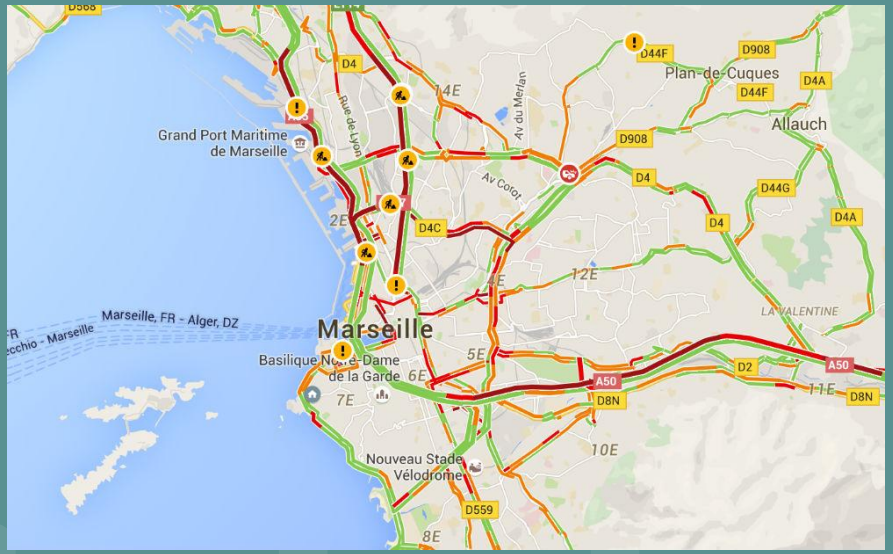
- Projet en collaboration avec l'ENTPE : simulateur de trafic routier
- Objectif : fournir des outils de visualisation rapides et intuitifs

Extraction du XML

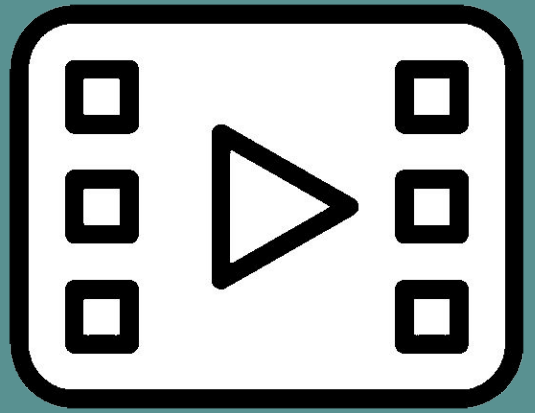
- Données sous forme d'arbre
  - <INSTANTS>
  - <VEHS>
  - <TRONCONS>

Deux types de visualisation

Statique



Dynamique

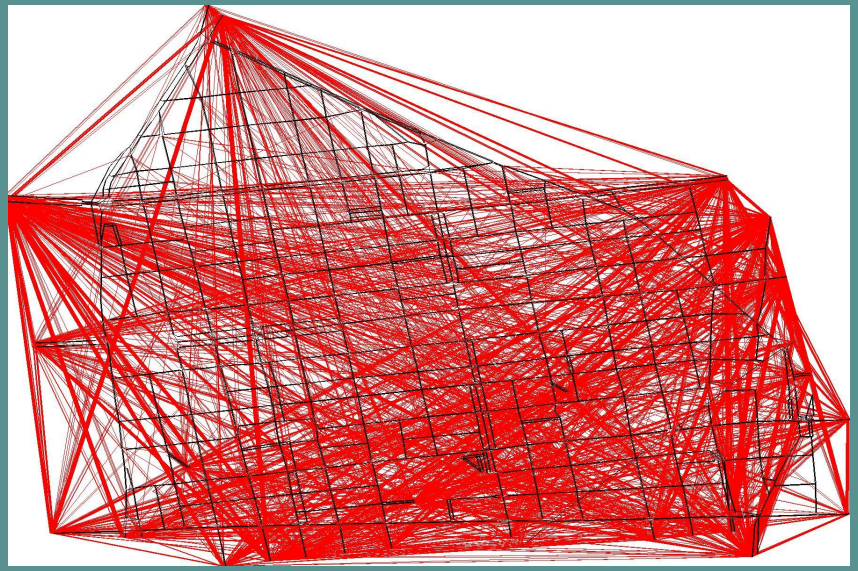


STATIQUE

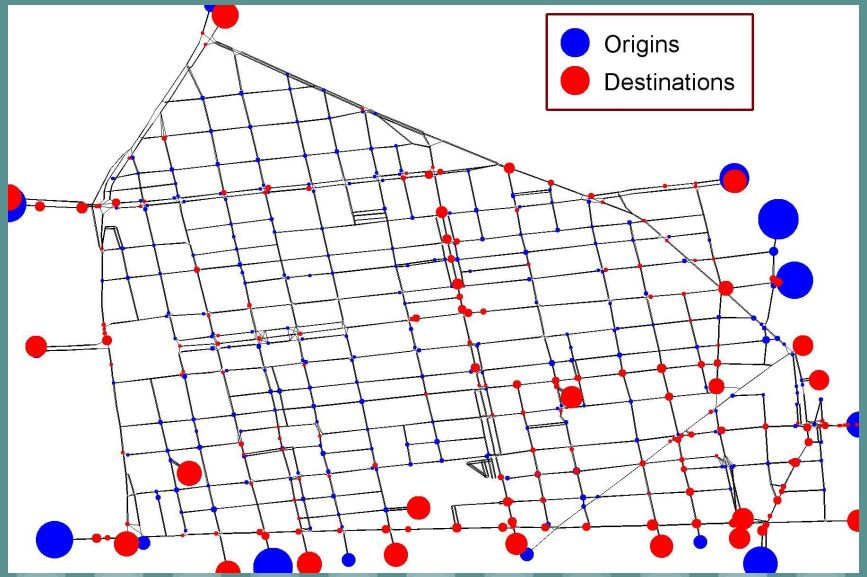
Statique

- Limite de vitesse
- Code couleur intuitif
- Aide à comprendre les trajets empruntés dans la simulation

Statique : Cartes OD



- Visualise de manière efficace les entrées et sorties du réseau
- Pas de représentation du trajet

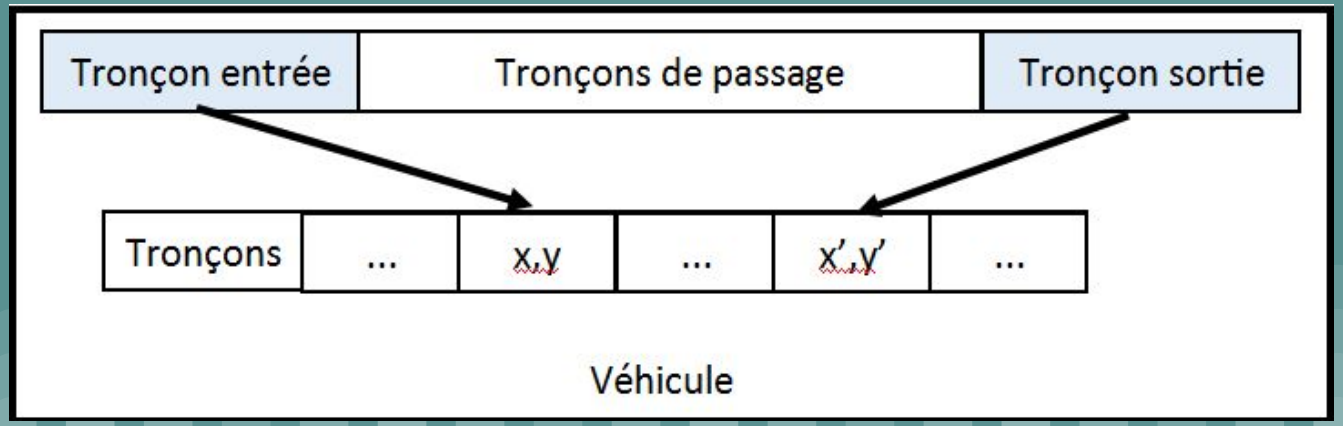


Statique : Extraction du XML

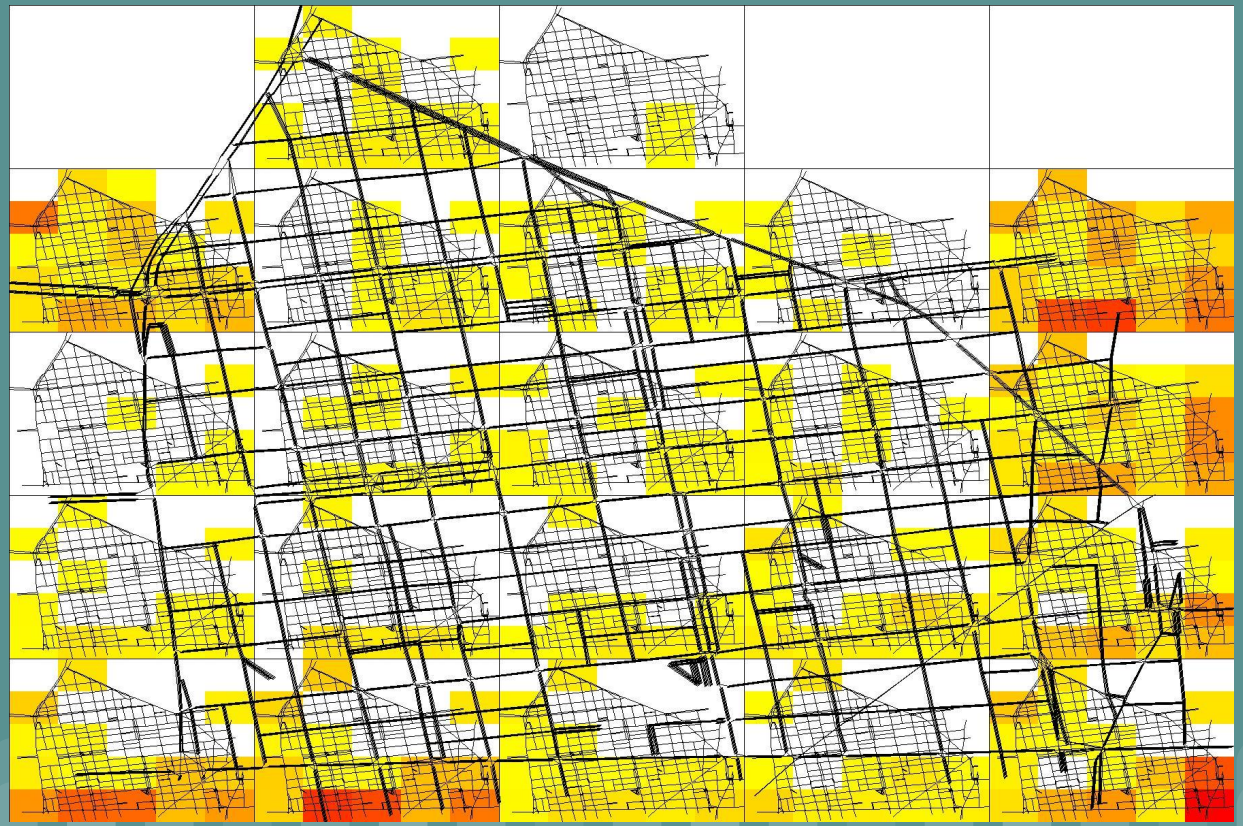
- Noeuds <VEHS> et <TRONCONS>

{ id du véhicule : liste des tronçons parcourus }

couplage avec la liste des tronçons pour accéder à la position

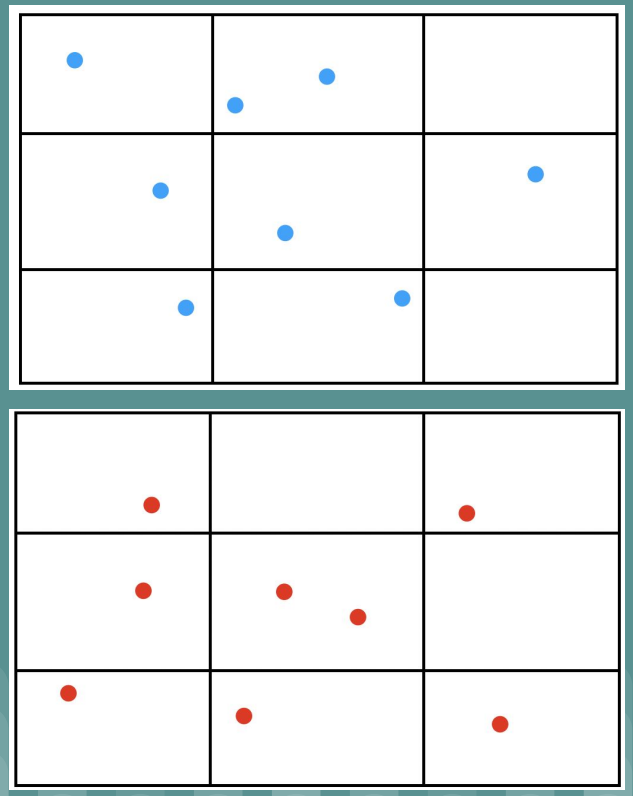


Statique : Cartes OD



- Carte OD à “cartes internes”
- Principe :
  - Découpage en secteurs
  - premier plan = origines
  - arrière plan = destinations
- Lecture non intuitive au premier abord

Statique : Cartes OD



Objectifs :

- ➔ Discrétiser la carte en secteurs
- ➔ Regrouper les origines et destinations par secteurs
- ➔ Créer une échelle de couleurs basée sur le maximum

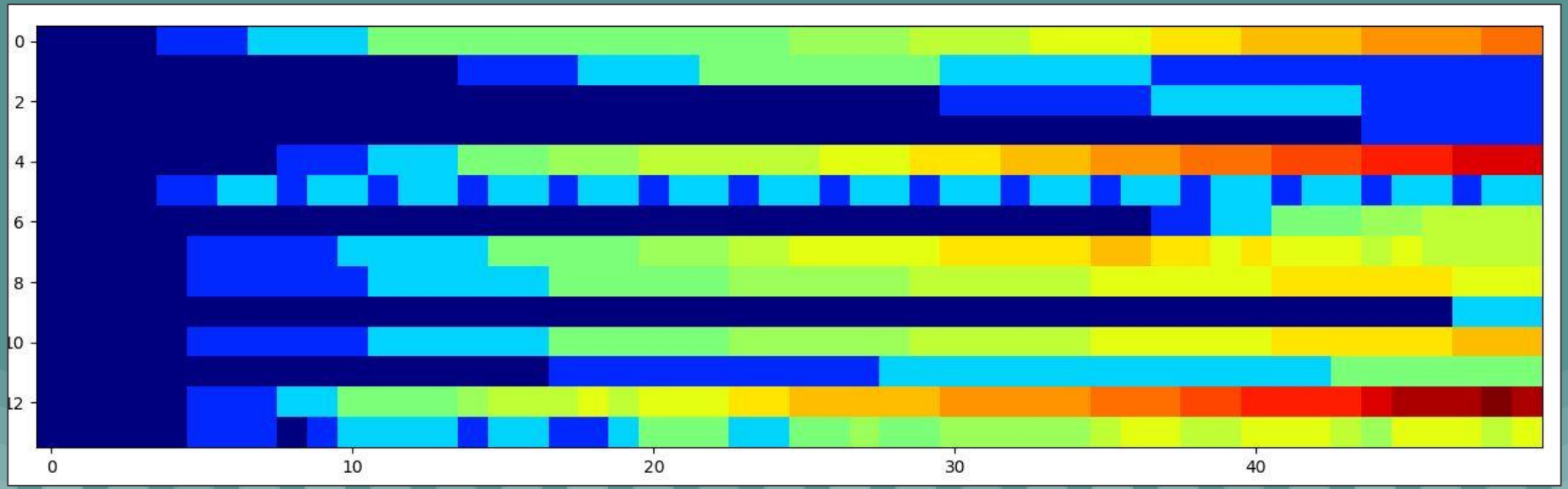
Statique : Cartes OD

```
def colorize(imag,xmin,xmax,ymin,ymax,color):
    for y in range(ymin,ymax):
        #print(y)
        for x in range(xmin,xmax):
            imag.putpixel((x, y), color)

def matrix_max(mat):
    m=-1e30
    for i in mat:
        for j in i:
            if j > m:
                m=j
    return m

def is_all_blank(l):
    for e in l:
        if e[:-1] != (255,255,255):
            return False
    else:
        return True

def scl(pt):
    X,Y = pt
    eps = 1
    Xmin,Xmax,Ymin,Ymax = (842872.935922 - eps, 844593.286 + eps, 6519780.692- eps, 652145
    x = int((X-Xmin)/(Xmax-Xmin)*w)
    y = int((Ymax-Y)/(Ymax-Ymin)*h)
    return (x,y)

def main(n):
    global w
    global h
    global Xmin
    global Xmax
    global Ymin
    global Ymax
    global transp
    transp = 55
    Xmin,Xmax,Ymin,Ymax = (842872.935922, 844593.286, 6519780.692, 6521456.1002)
    w,h = (1800,1200)
    color = (255,255,255,255)
    global im
    global draw
    im = Image.new('RGBA', (w,h), color)
    draw = ImageDraw.Draw(im)
    odmap = ODMap('Bulle',n,n,[[]])
    odmap.create_reduced_matrix(ExtractODMat.main())
    # print(odmap.reduced_matrix)
    odmap.display()
    im.save('ODmap {}x{}'.format(n,n),'JPEG')
```

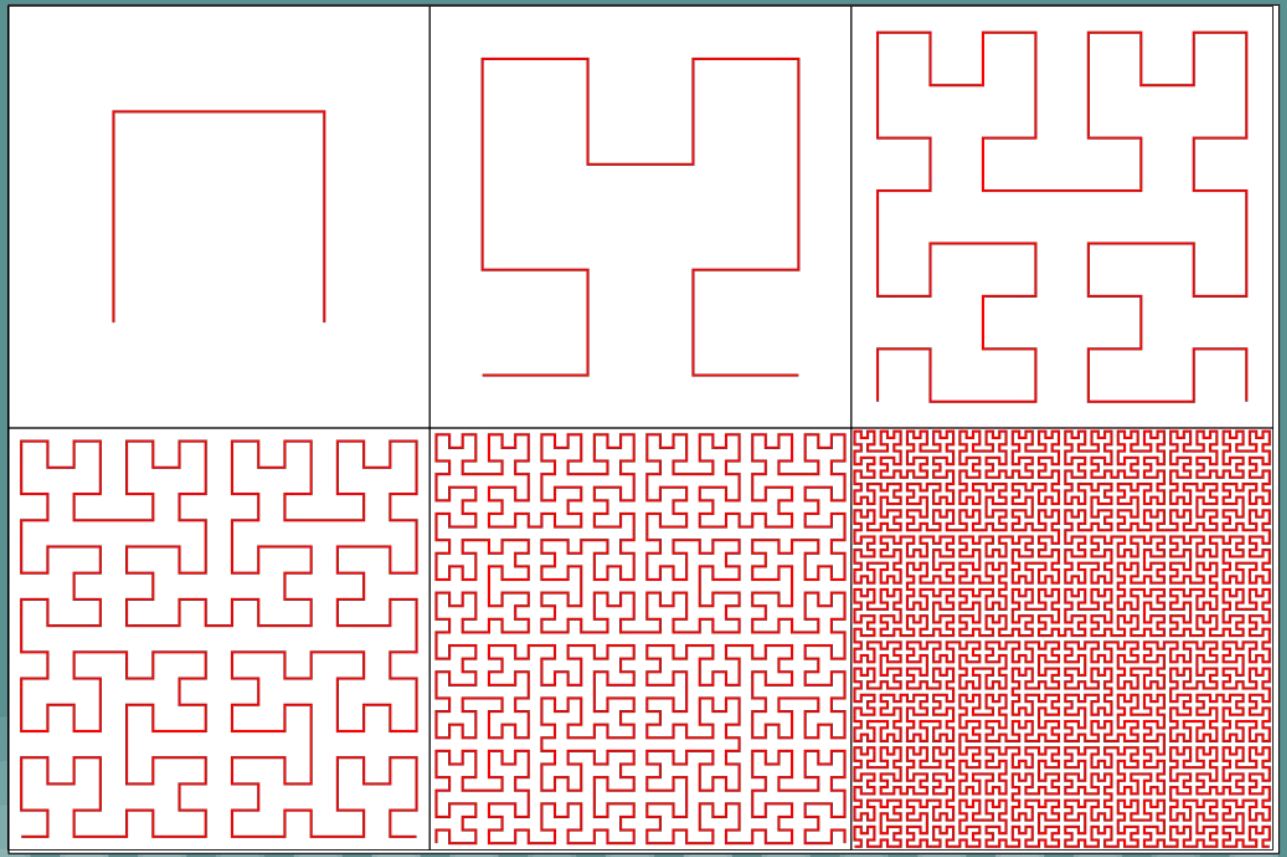
- 3 classes : ODMap, OMap et DMap. ODMap -> OMap -> [DMap]
- Principe :
  - Récupération de tous les couples O-D (ExtractODMat.main())
  - Création d'une matrice réduite (create_reduced_matrix())
  - Création de la visualisation (display())
  - Sauvegarde du résultat sur la machine

Motion Rugs : Présentation

- Visualisation statique d'un système dynamique
- Idée nouvelle
- Passer de la 2D à la 1D

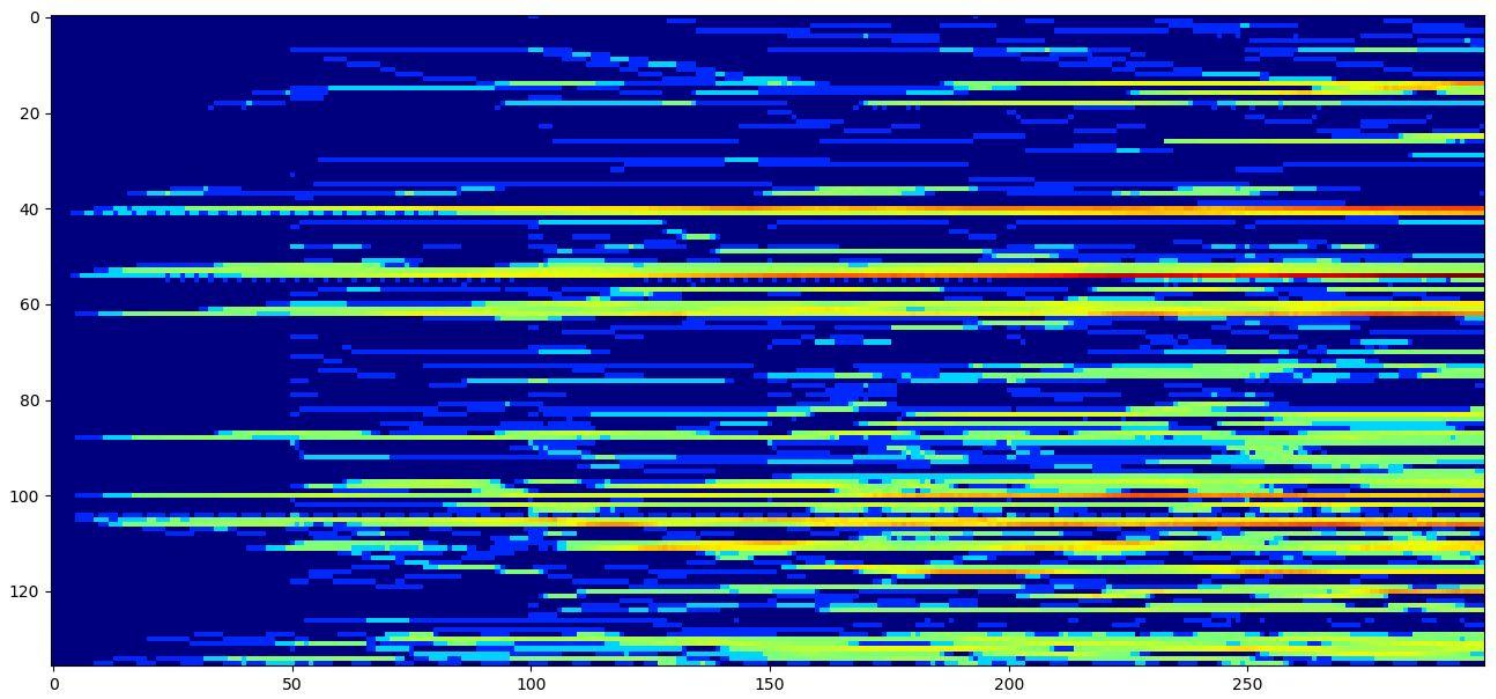


Motion Rugs : De la 2D à la 1D



Motion Rugs : Conclusion

- Difficilement lisible
- Passage de la 1D à 2D



DYNAMIQUE

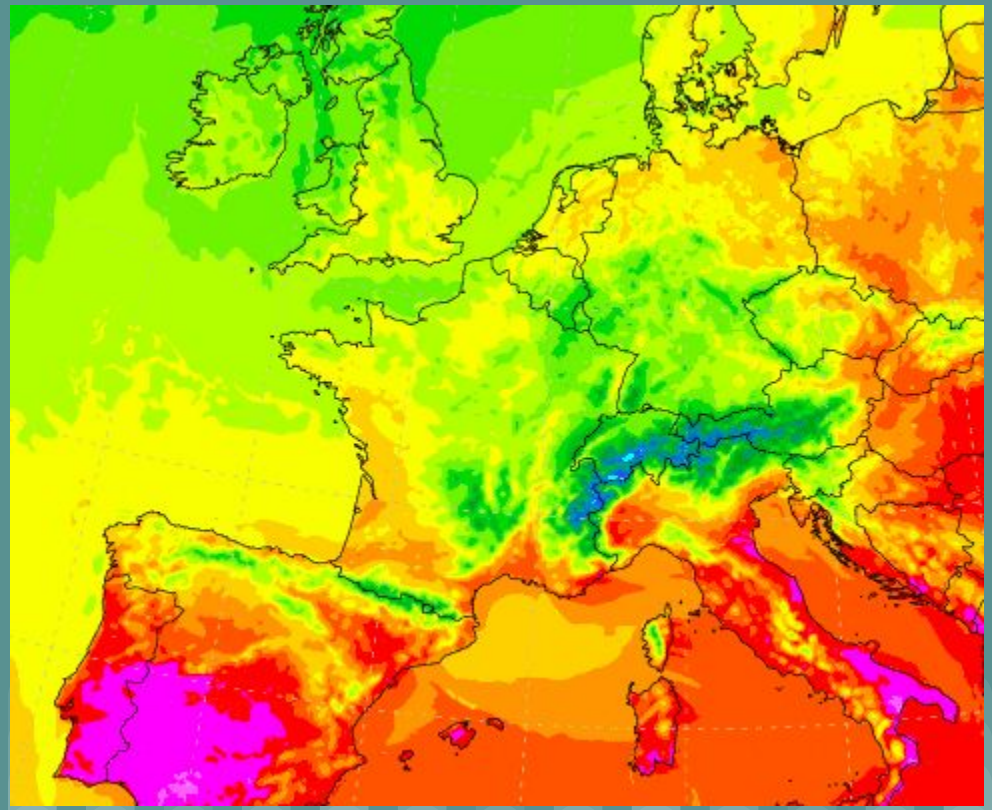
Dynamique : Extraction du XML

- Noeud <INSTANTS>
- Liste des véhicules présents sur le réseau avec les noeuds <TRAJ>
- Chaque véhicule a des informations
    - <TRAJ abs= acc= dst= id= ord= tron= type= vit= voie= z= />
- On s'intéresse uniquement à abs et ord qui nous permettront de placer le véhicule sur la carte à l’instant t

Dynamique : Replay

- Chaque image est le résultat du placement de tous les véhicules sur le réseau à l’instant considéré
- Vitesse de défilement paramétrable
- Volumineux en espace disque

Dynamique : Flux de Chaleur



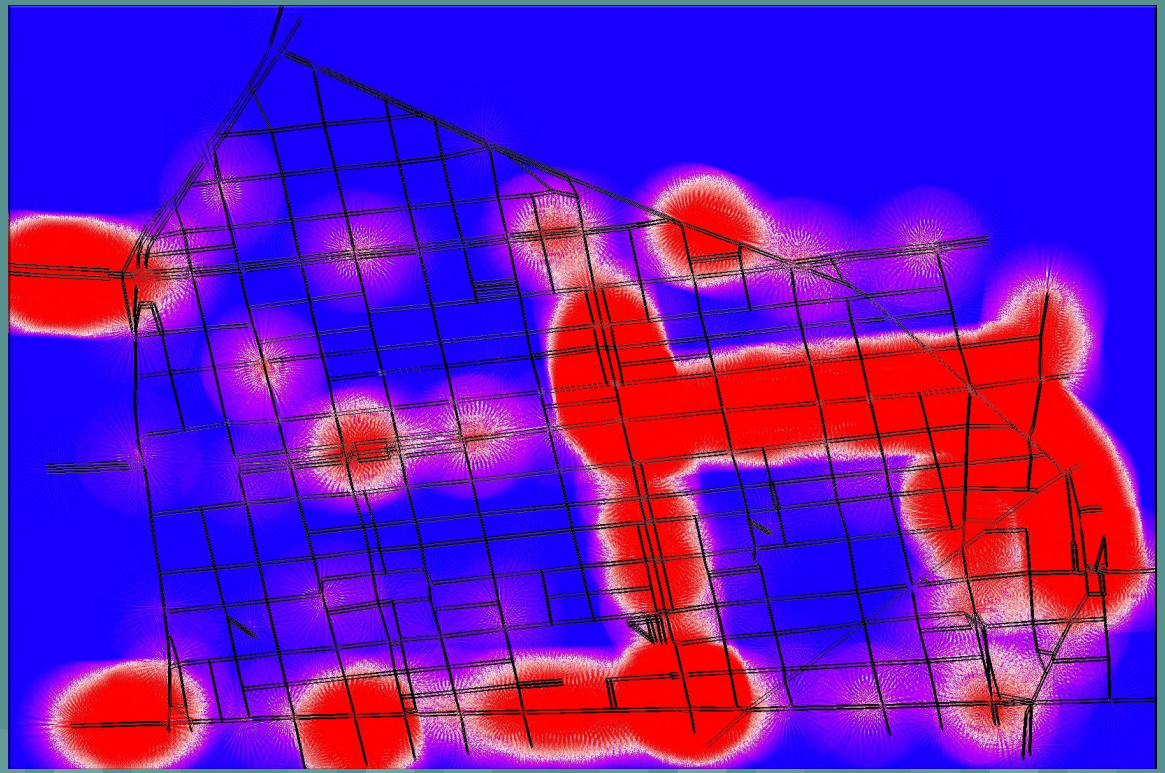
- Animation vidéo
- Nuage rouge où il y a plus de véhicules, bleu où il en a moins
- Objectif : avoir un aperçu global des zones de congestion avec une vidéo courte

Dynamique : Flux de Chaleur

L’idée est de créer des zones de chaleur autour de chaque véhicule. La décroissance de la “chaleur” est exponentielle avec un paramètre α fixé arbitrairement à 0,02. On affecte ensuite à la coordonnée (x,y) une couleur qui dépend uniquement de la fonction :

$$f_t(x,y) = \sum_{i}^{n} e^{-\alpha \cdot d_i(x,y)}$$

Dynamique : Flux de Chaleur



- 1500ème instant
- Idée globale instantanée des zones de congestion
- TRÈS long à calculer

Conclusion

- Grande diversité de résultats

- Globalement exploitable

- Améliorations possibles :
  - Optimisation du temps de calcul, des programmes…
  - Autres visualisations et couplage avec des données réelles