inventeurs du monde numérique

Perception pour les Véhicules autonomes

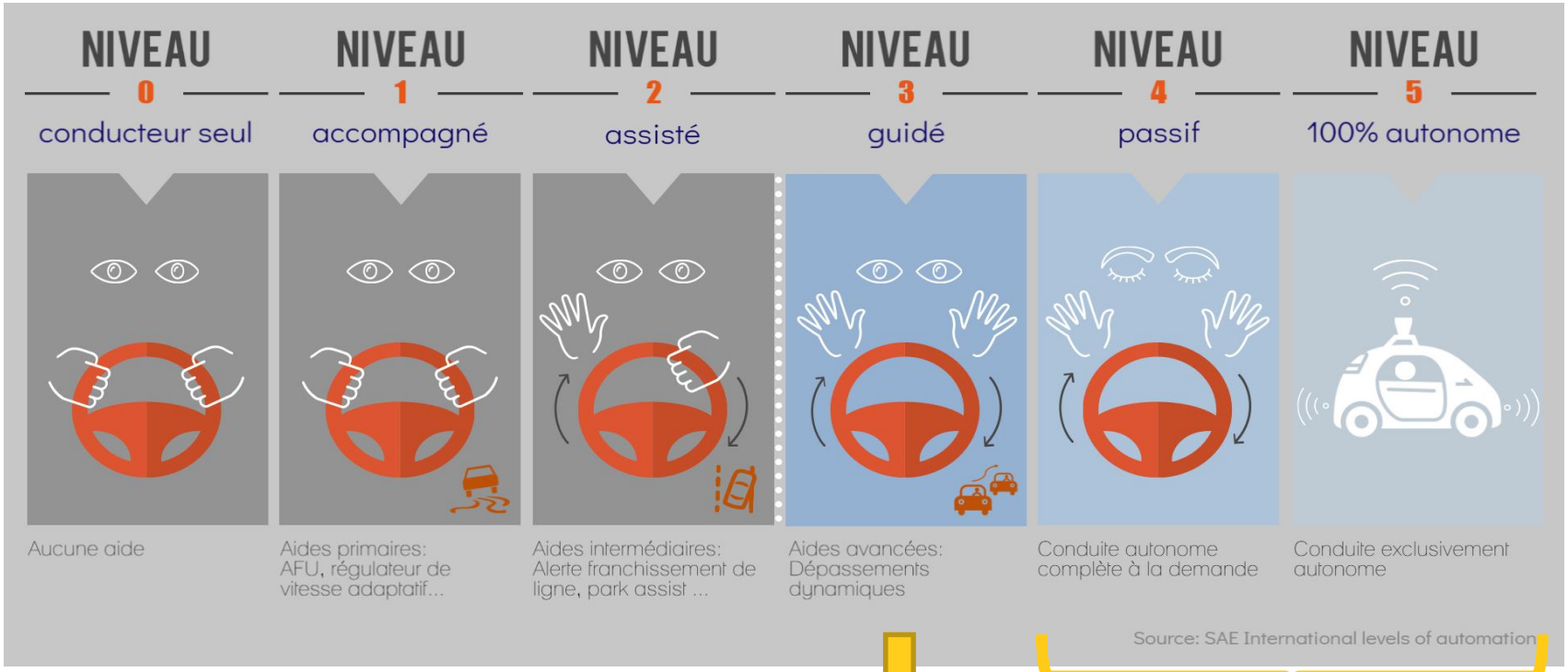
Présentation ISN 2018

N. Turro & équipe Chroma

01

Introduction aux véhicules autonomes

Véhicules Autonomes, de quoi parle-t-on?



Largement commercialisé

TESLA

R & D

WAYMO

Environnement maîtrisés :

easymile

navya be fluid

Inria inventeurs du monde numérique

État des lieux

Février 2018 : Waymo : 5 millions de km, Uber 3.2 millions, PSA : 150000 (voies rapides)

Decembre 2017, rapport : "Report on Autonomous Mode Disengagements For Waymo Self-Driving Vehicles in California "30 véhicules pour 563200 km parcourus

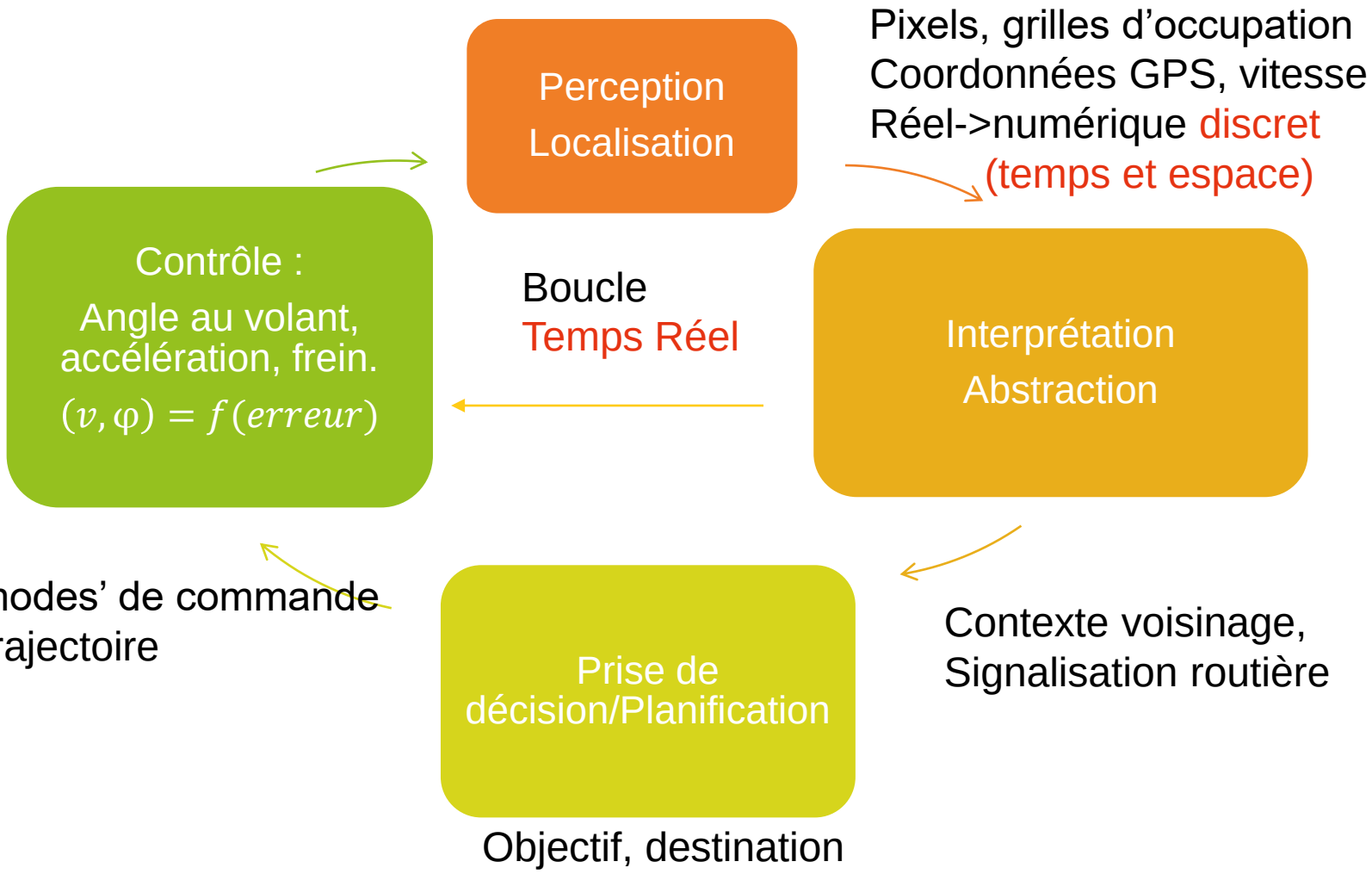
- 63 interventions humaines :
- 6 sur autoroute, 57 dans rues
- Problèmes prédominants :

  panne matérielle, incertitudes de perception, configuration du véhicule et de son environnement
- Valeo, Nvidia, Uber : > 1 reprise en main tous les 10km

Accidents mortels :

- 1 mort tous les 100Millions km en conduite humaine
- Tesla autopilot 1er mort a 130Million km, 2eme récemment
- Uber, 1 mort en 3.2 millions…

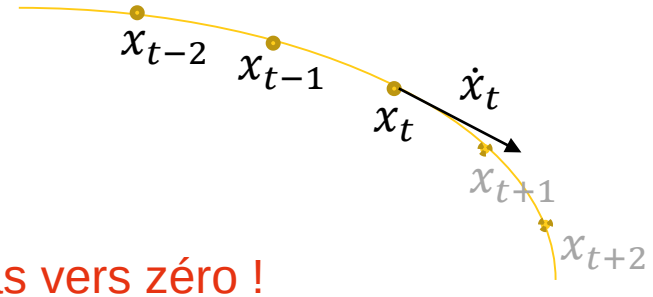
Comment cela fonctionne ?



Perception numérique : discrétisation du temps

Continu

$$\begin{cases} \dot{x}_t = \lim\limits_{dt \to 0} \left(\frac{x_t - x_{t-1}}{dt}\right) \\ \ddot{x}_t = \lim\limits_{dt \to 0} \left(\frac{\dot{x}_t - \dot{x}_{t-1}}{dt}\right) \end{cases}$$



Discret

$$\begin{cases} \dot{x}_t = \frac{x_t - x_{t-1}}{\Delta t} \\ \ddot{x}_t = \frac{\dot{x}_t - \dot{x}_{t-1}}{\Delta t} = \frac{x_t - 2x_{t-1} + x_{t-2}}{(\Delta t)^2} \end{cases}$$

Ne tend pas vers zéro !

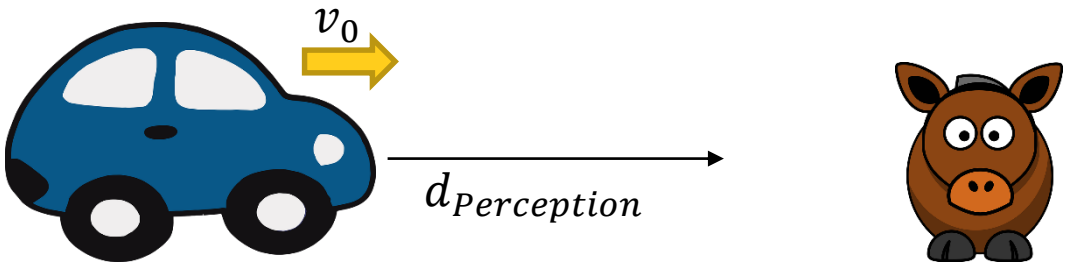
Passé !

- Filtrage = retard
  Interpolations (linéaire, cercle, bézier) pour valeurs intermédiaire, rayon de courbure…
  Même avec des mesures parfaites, nos calculs seront approximatifs

Notion de Temps réel

En informatique, on parle d'un **système temps réel** lorsque ce système est capable de contrôler (ou piloter) un procédé physique à une vitesse adaptée à l'évolution du procédé contrôlé

Etant donné un capteur qui perçoit des véhicules à 10Hz à 50m quelle vitesse est-il prudent de se déplacer ? Relation avec le temps de calcul t_c



On veut : $d_{arrêt} < d_{Perception}$

$d_{arrêt} = v_0 t_{réaction} + d_{freinage}$

$d_{freinage} = \frac{-{v_0}^2}{2a_0}$

$t_{réaction} = t_{perception} + t_{calcul} = \frac{1}{f_p} + t_c$

$\Rightarrow \quad \frac{{v_0}^2}{2a_0} + v_0(\frac{1}{f_p} + t_c) < d_{Perception}$

$\Rightarrow$

$a_0 = -10\ ms^{-2}$; $f_p = 10Hz$

$d_{Perception}$ = 50m

$t_c = \frac{1}{f_p} = 0.1s \Rightarrow v_0 <$ 107 km/h

$t_c = \frac{10}{f_p} = 1s \Rightarrow v_0 <$ 81 km/h

02

Perception et fusion de capteurs

Perception, un des enjeux majeurs

Accident Tesla, Uber

Capteurs

Proprioceptifs : GPS, odométrie, centrale inertielle

Extéroceptifs

Capteur	Défauts
Caméras	Sensibles au conditions d'éclairage
Lidars	Résolution verticale/prix, vitres, calibration
Radars	Résolution angulaire
Ultra-sons	Portée, résolution
Communication	Limitée infrastructure/véhicules

Pas de capteur idéal, combiner l'information !

Fusion de capteur simple : localisation

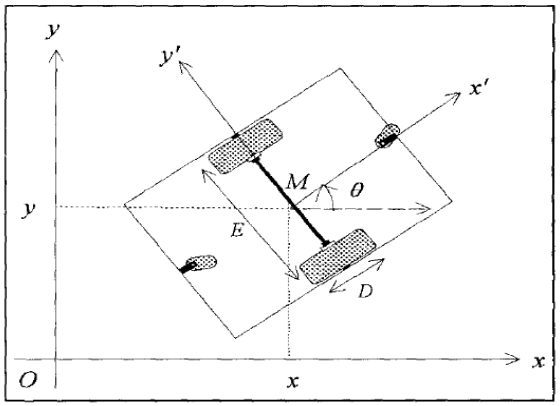
3 capteurs proprioceptifs imparfaits

Fusion

Etat simple : position du véhicule

GPS : Système satellitaire (GNSS, GLONASS, GALILEO) + EGNOS

Odométrie :



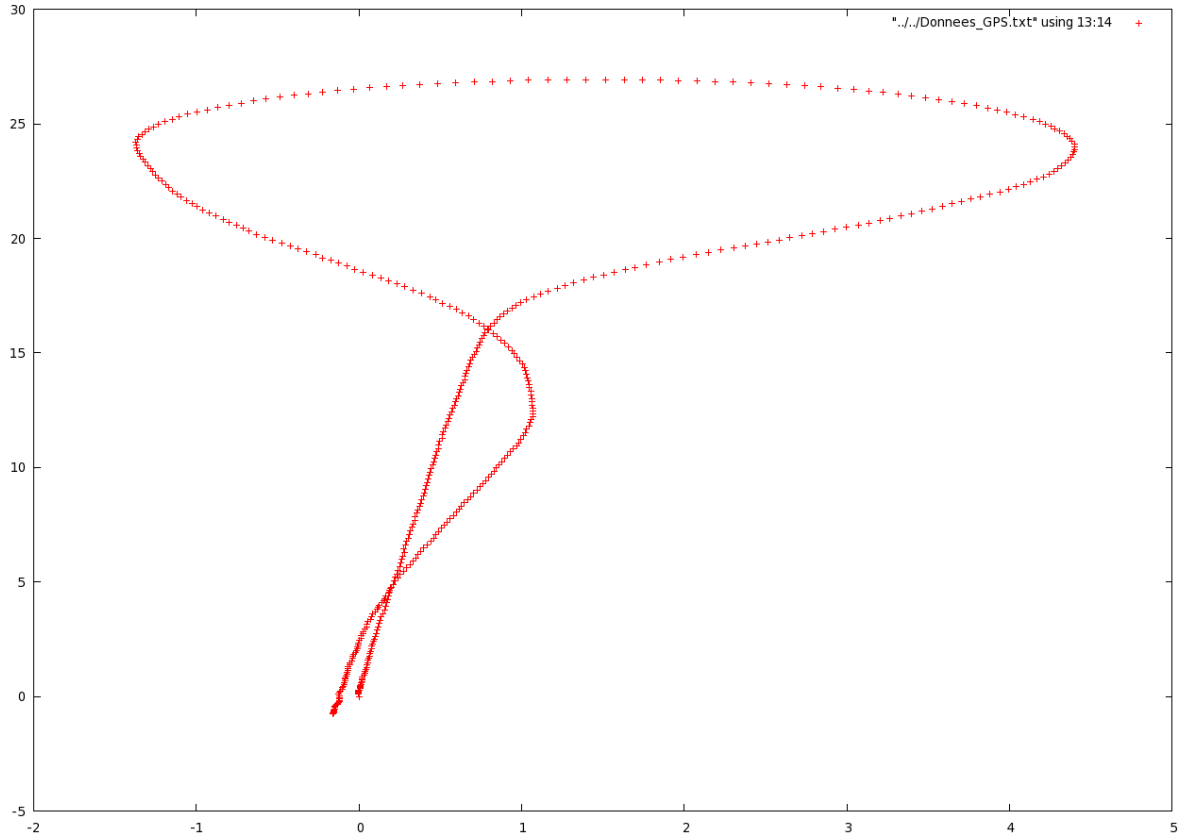
$$X_{n+1} = X_n + \frac{S_R + S_L}{2}\cos(\theta_n)$$

$$Y_{n+1} = Y_n + \frac{S_R + S_L}{2}\cos(\theta_n)$$

$$\theta_{n+1} = \theta_n + \frac{S_R - S_L}{Largeur\ de\ l'essieu}$$

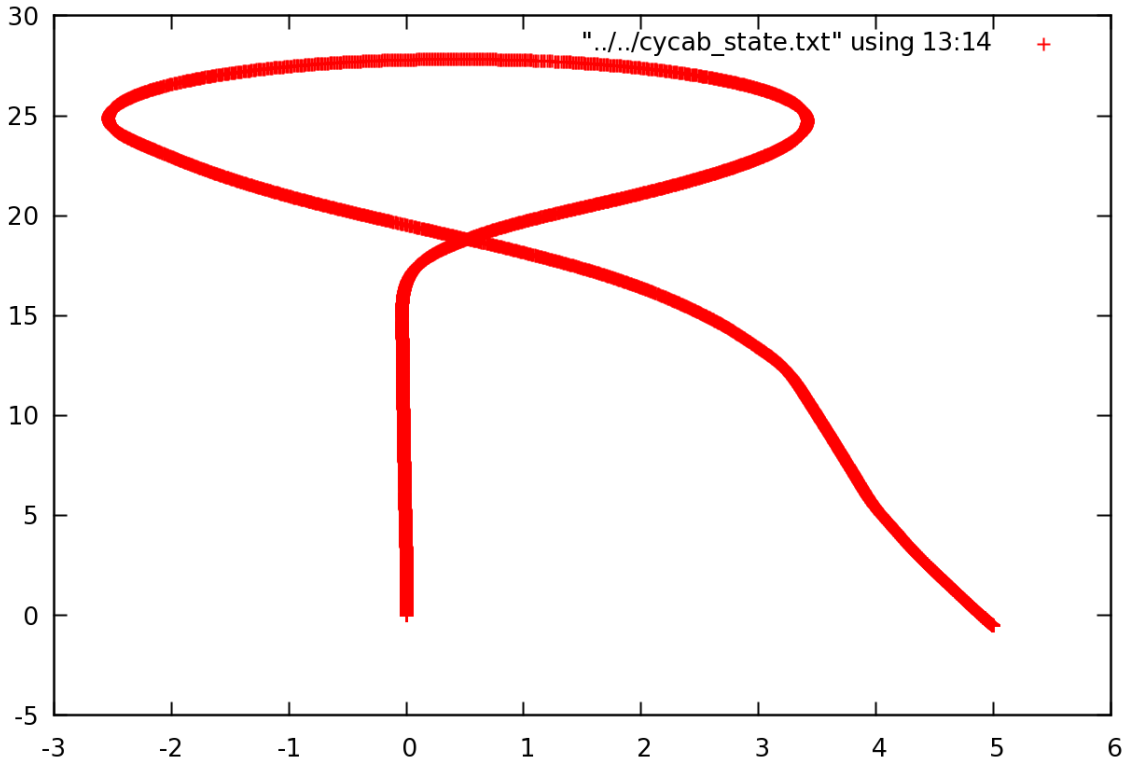
Centrale inertielle : mesure des accélérations linéaires et angulaires

Fusion de capteur pour la localisation : GPS



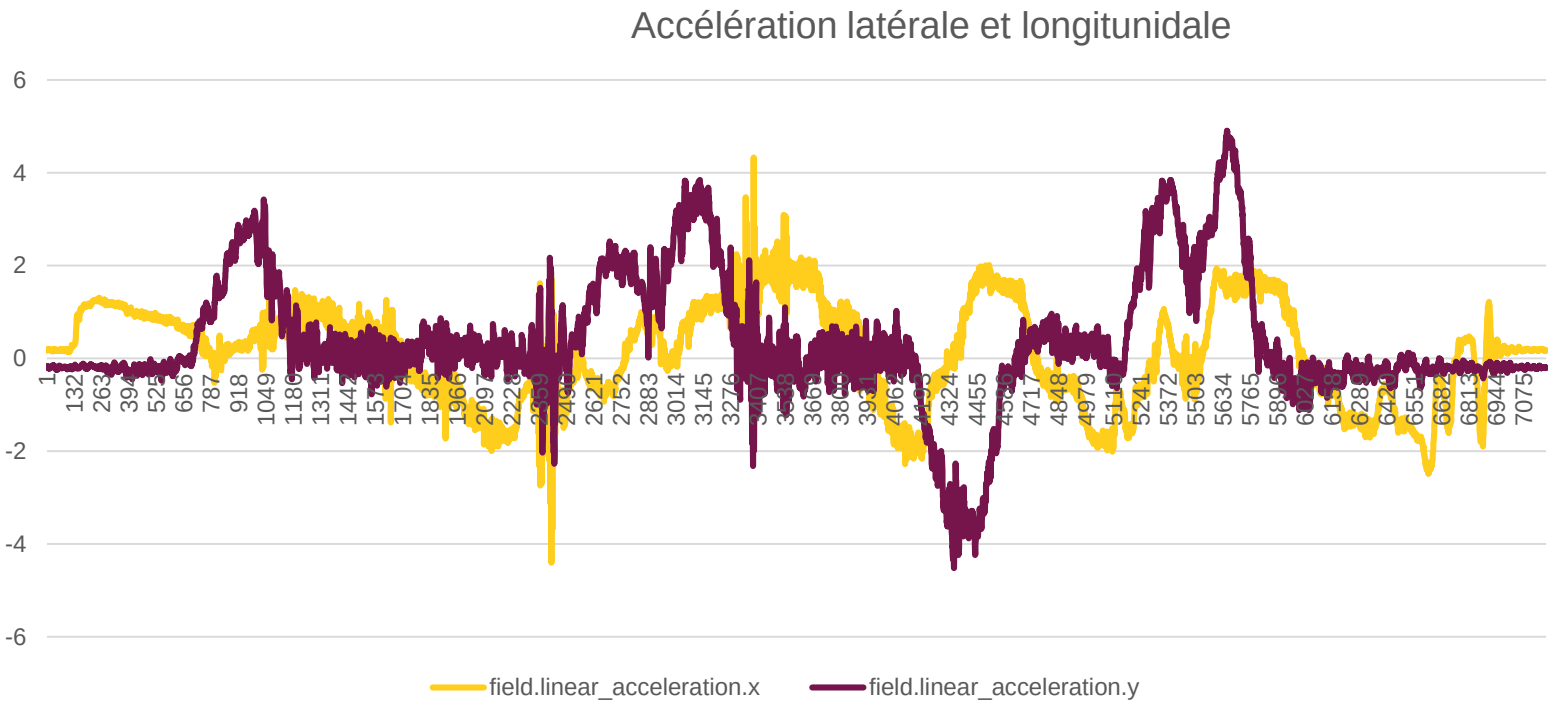
- Fréquence : 1Hz, 28m à 100km/h
- Tunnels
- Bruit/Sauts

Fusion de capteur pour la localisation : Odométrie



Glissement
Dérive angulaire

Fusion de capteurs pour la localisation : Centrale inertielle

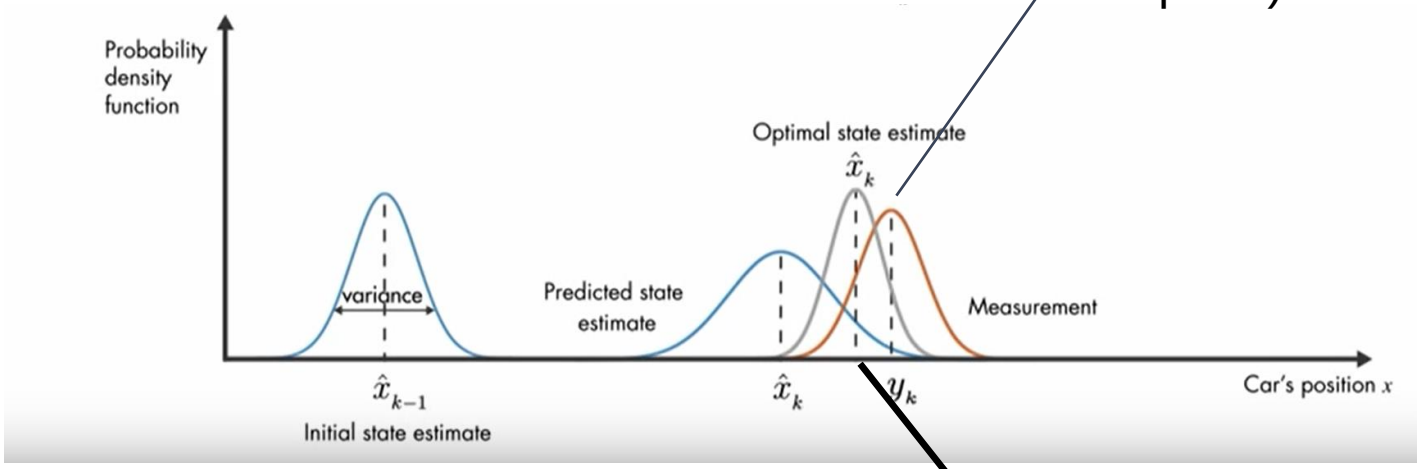


Bruit, BIAIS d'installation, dérivation double !

inventeurs du monde numérique

Incertitudes et modèles pour filtrage de Kalman

Variance de la mesure (calibration capteur)



Modèle physique/dynamique
+ contrôle
+ 'bruit'

Estimation optimale avec variance réduite

Fusion de capteur pour la localisation : Filtre de Kalman

Prédiction :

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k \mathbf{u}_k \quad \text{(état prédit)}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k \quad \text{(estimation prédite de la covariance)}$$

Entrées, mesures

Mise à jour :

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1} \quad \text{(innovation)}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k \quad \text{(covariance de l'innovation)}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \quad \text{(gain de Kalman } \textit{optimal}\text{)}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \quad \text{(état mis à jour)}$$

$$\mathbf{P}_{k|k} = (I - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \quad \text{(covariance mise à jour)}$$

Sorties

avec

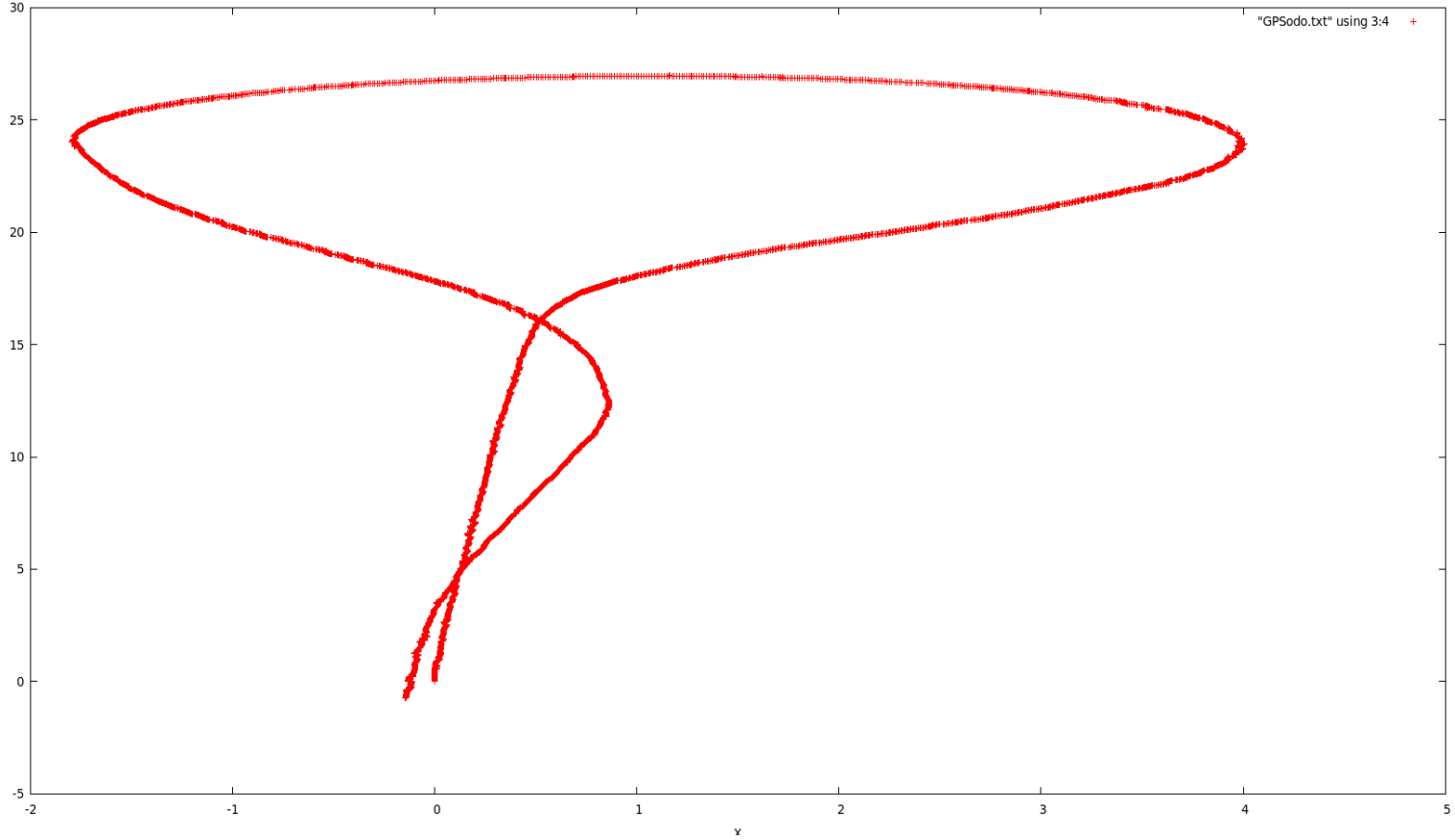
- $\mathbf{F}_k$: matrice qui relie l'état précédent k-1 à l'état actuel k
- $\mathbf{u}_k$: entrée de commande
- $\mathbf{B}_k$: matrice qui relie l'entrée de commande u à l'état x
- $\mathbf{P}_{k|k-1}$: matrice d'estimation *a priori* de la covariance de l'erreur
- $\mathbf{Q}_k$: matrice de covariance du bruit du processus (en anglais *process noise*)

avec

- $\mathbf{z}_k$: observation ou mesure du processus à l'instant k
- $\mathbf{H}_k$: matrice qui relie l'état $\mathbf{x}_k$ à la mesure $\mathbf{z}_k$
- $\mathbf{P}_{k|k}$: matrice d'estimation *a posteriori* de la covariance de l'erreur
- $\mathbf{R}_k$: matrice de covariance du bruit de mesure
- I : matrice identité aux dimensions adéquates

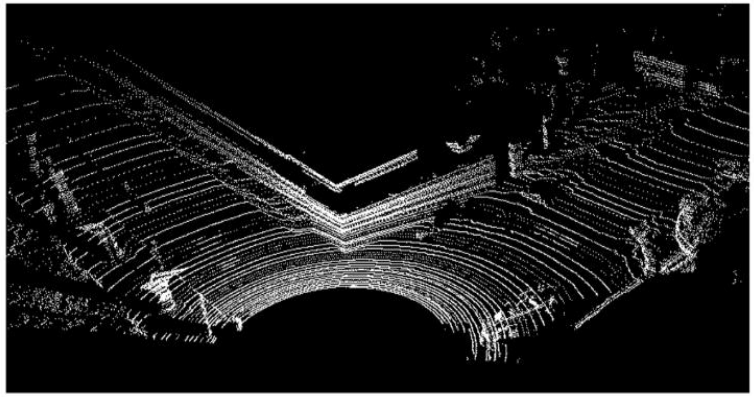
$$X_k = \begin{bmatrix} X \\ Y \\ X' \\ Y' \end{bmatrix} \qquad F_k = \begin{bmatrix} 1.0 & 0.0 & dt & 0.0 \\ 0.0 & 1.0 & 0.0 & dt \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

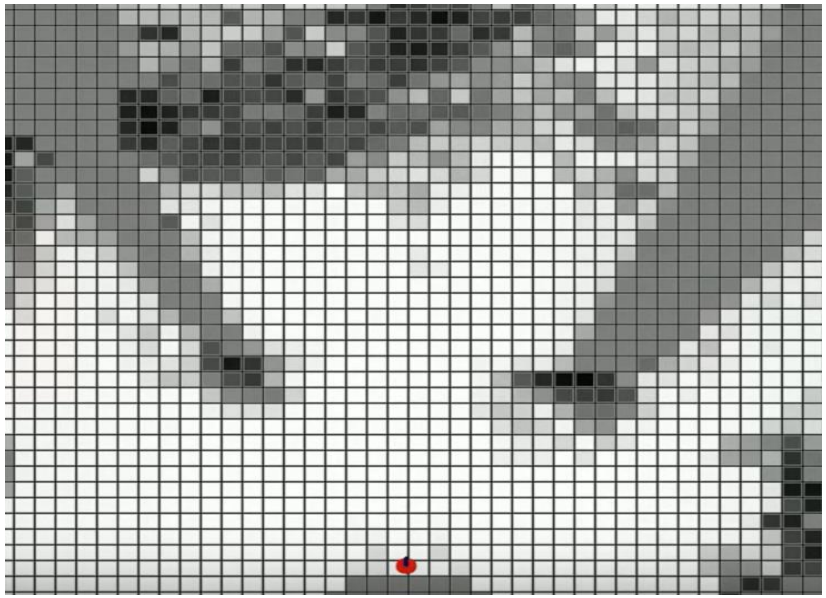
Fusion de capteur pour la localisation : Odométrie + GPS par filtrage de Kalman



Grille d’occupation Bayésienne

(b) 3D point cloud





Grille vue de dessus
Valeur = probabilité d’occupation

0 = blanc = libre
1 = noir = occupé
0.5 = gris = on ne sait pas (pas observé..)
Représentation commune à tous les capteurs !

Filtrage et fusion Bayésienne

$$\mathbf{P}(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\bar{A})P(\bar{A})} :$$

fait

Observation

Fiabilité la detection (y compris fausses alertes)

Thèse C. Coué (Inria e-motion) :

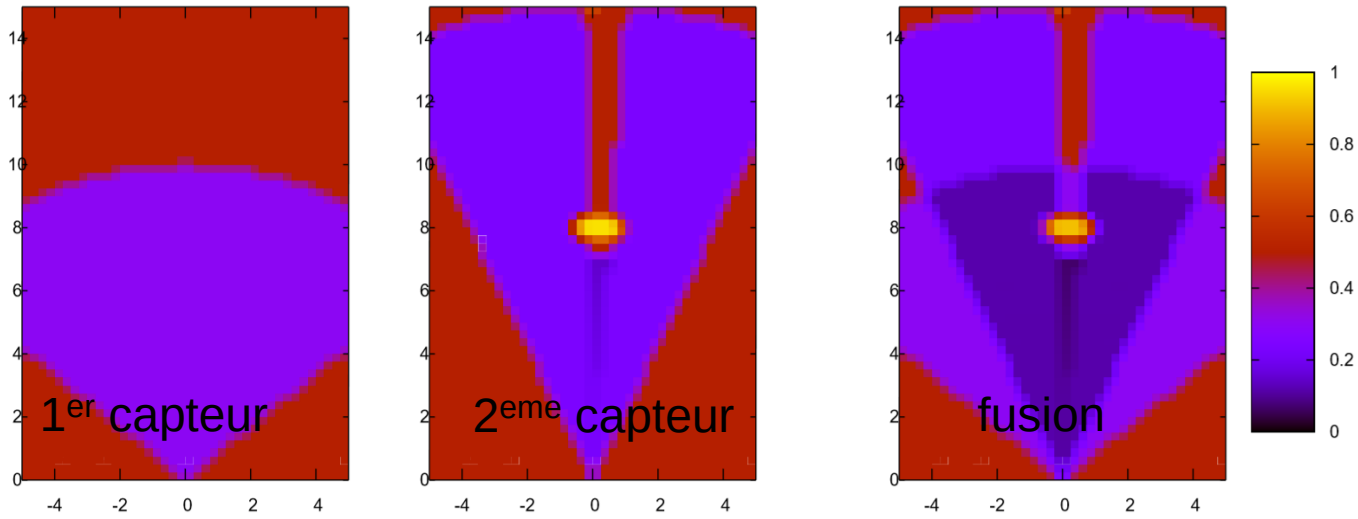
Pour S capteurs :

Connaissance à priori

observation

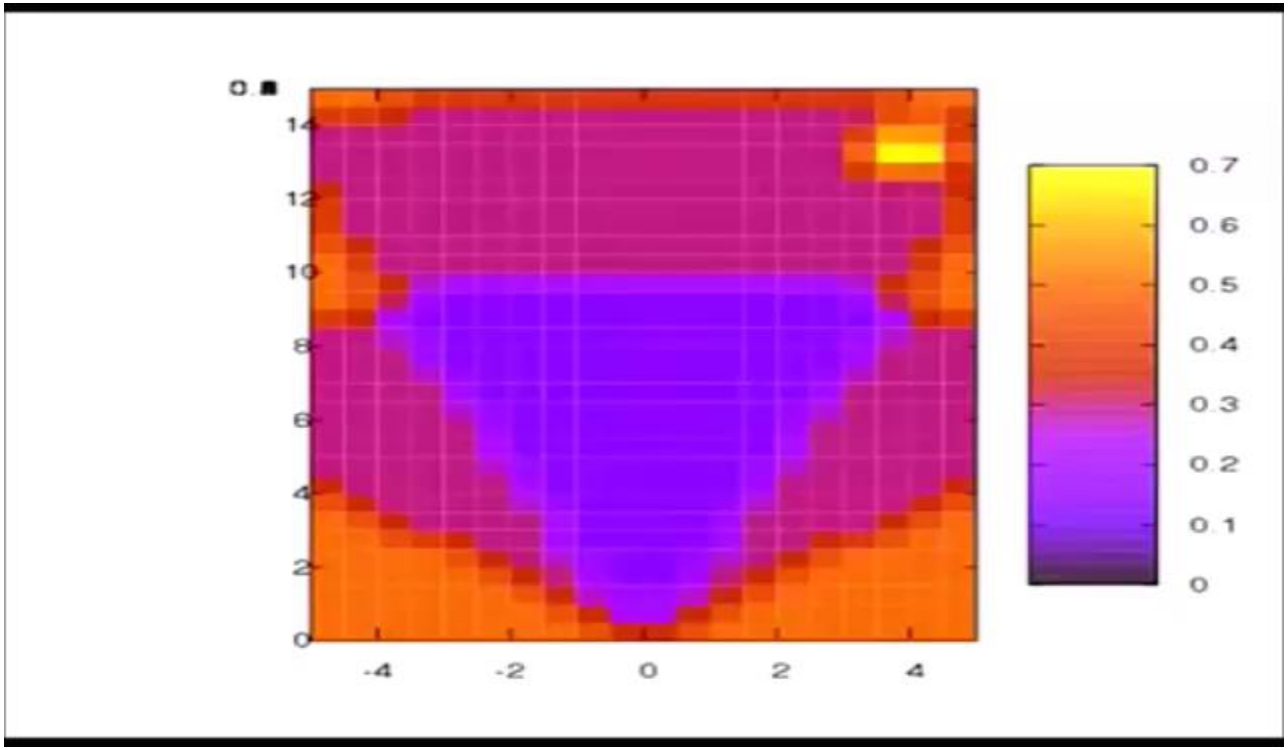
$$\mathrm{P}(E_x \mid z_1 \ \ldots \ z_S \ x) = \frac{\mathrm{P}(E_x \mid x) \prod_{s=1}^{S} \mathrm{P}(z_s \mid E_x \ x)}{\sum_{E_x} \left(\mathrm{P}(E_x \mid x) \prod_{s=1}^{S} \mathrm{P}(z_s \mid E_x \ x) \right)}$$

Fusion de grilles d’occupation



Fusion de grilles d’occupation

Fusion ‘temporelle’ (prédiction/observation/mise à jour)

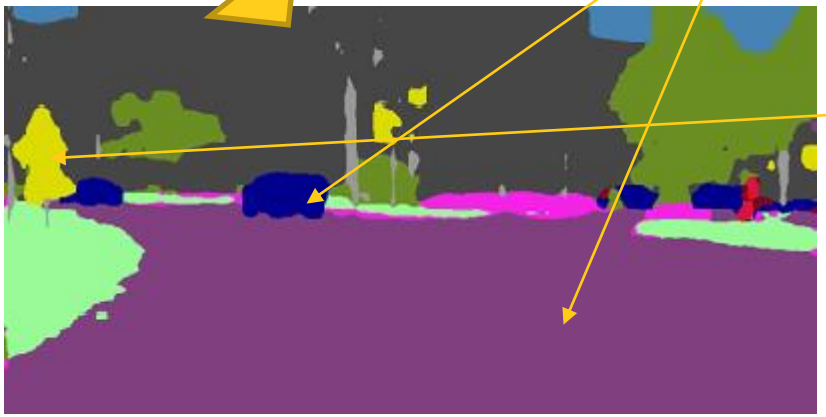


03

Interprétation, abstraction et apprentissage profond

Réseaux de neurones convolutifs pour la perception

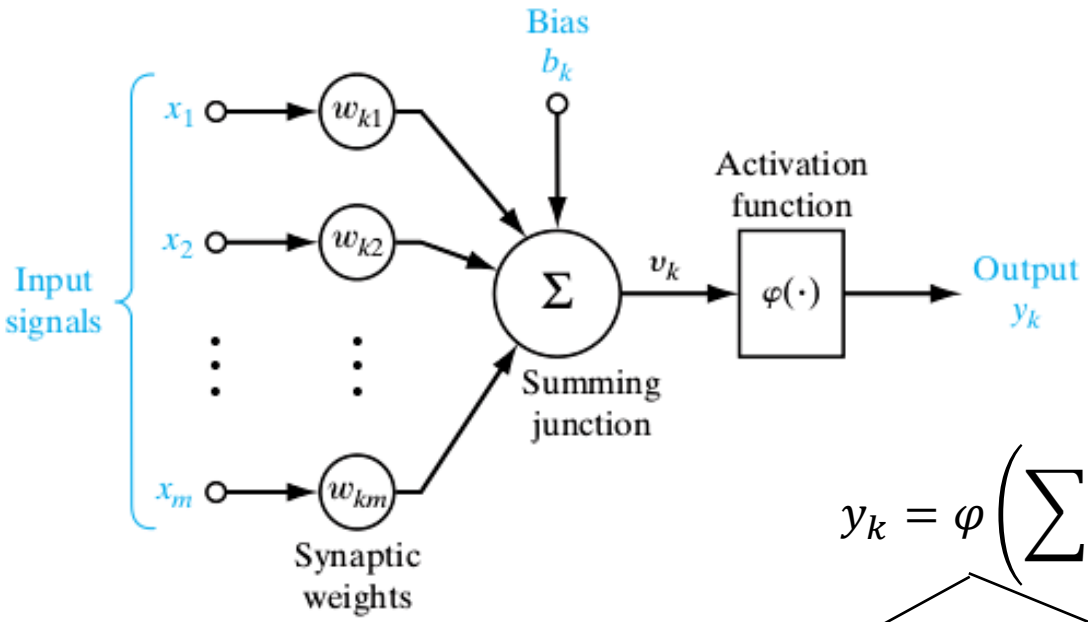
CNN



Interprétation
Abstraction

Group	Classes
flat	road · sidewalk · parking[+] · rail track[+]
human	person[*] · rider[*]
vehicle	car[*] · truck[*] · bus[*] · on rails[*] · motorcycle[*] · bicycle[*] · caravan[*+] · trailer[*+]
construction	building · wall · fence · guard rail[+] · bridge[+] · tunnel[+]
object	pole · pole group[+] · traffic sign · traffic light
nature	vegetation · terrain
sky	sky
void	ground[+] · dynamic[+] · static[+]

Réseaux de neurones convolutifs pour la perception
Un neurone



Paramètres à identifier

$$y_k = \varphi\left(\sum w_{k_i} x_i + b_i\right)$$

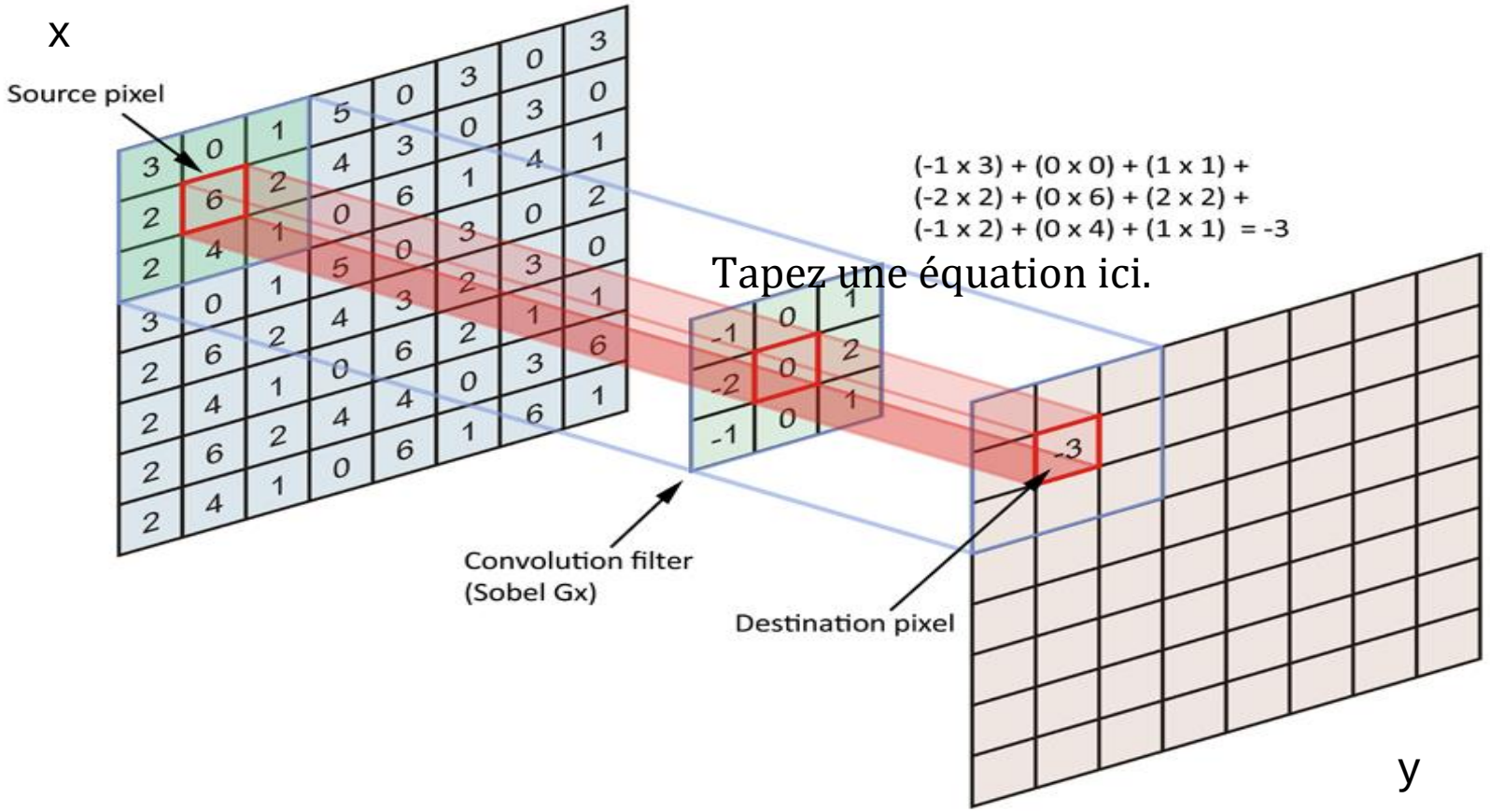
ReLu

$$f(x) = max(x, 0)$$

Softmax

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Réseaux de neurones convolutifs pour la perception
Convolution



Réseaux de neurones convolutifs pour la perception
Cas particulier, image et convolution

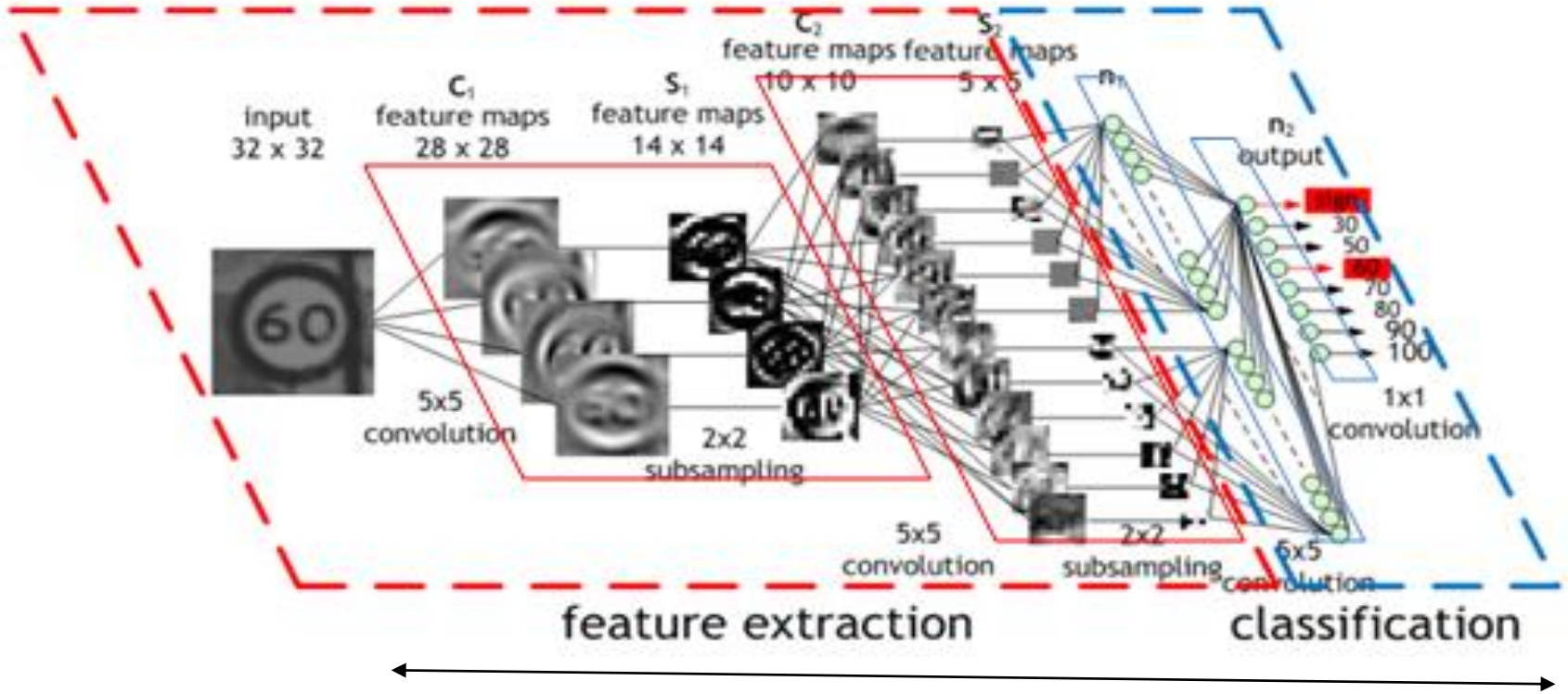
Détection de contours

Sobel

$$* \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} =$$

Réseaux de neurones convolutifs pour la perception

Réseau global



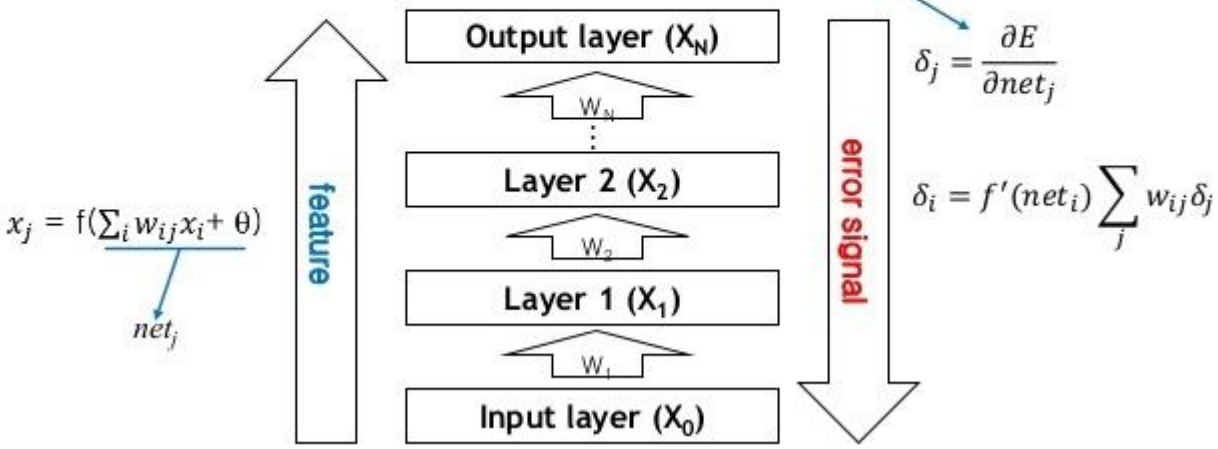
(apprentissage) PROFOND : millions de paramêtres

Réseaux de neurones convolutifs pour la perception

Apprentissage

- Gradient descent algorithm to minimize error E.

$$w_{ij} \leftarrow w_{ij} - \eta \frac{\partial E}{\partial w_{ij}} \qquad \frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial w_{ij}} = \delta_j x_i$$



Nécessite des milliers d'entrées classifiées pour l'apprentissage

Réseaux de neurones convolutifs pour la perception
La révolutions des années 2010+

Conjonction de :

- **Puissance de calcul : GPU, massivement parallèle**
- **Bases de données numériques :**

- **Frameworks informatiques puissants :**
  - Ex : Tensorflow, Keras

Réseaux de neurones convolutifs pour la perception

Keras

```
model = Sequential()

model.add(Convolution2D(32, 3, 3, activation='relu',
input_shape=(1,28,28)))
model.add(Convolution2D(32, 3, 3, activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

model.fit(X_train, Y_train,
          batch_size=32, nb_epoch=10, verbose=1)

score = model.evaluate(X_test, Y_test, verbose=0)
```

Abstraction :

Code de la route,

signalisation statique

signalétique dynamique (feux, travaux)

Priorités

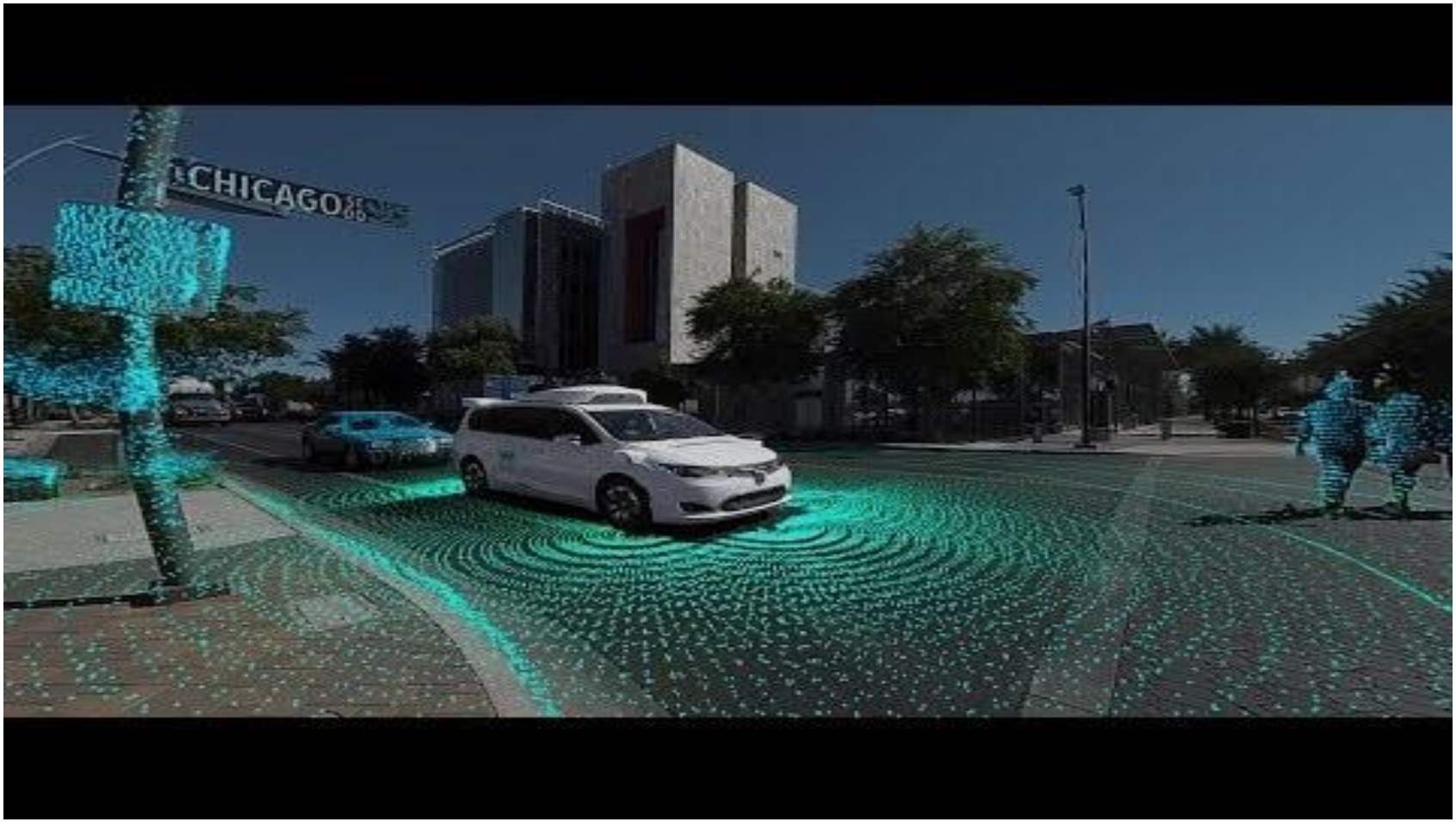
Raisonnement comportemental :

anticipation des actions des tiers, modèle comportemental

‘bon moment’ pour doubler ?

Modèles bayesiens, deep learning

Résultat



Evolutions :

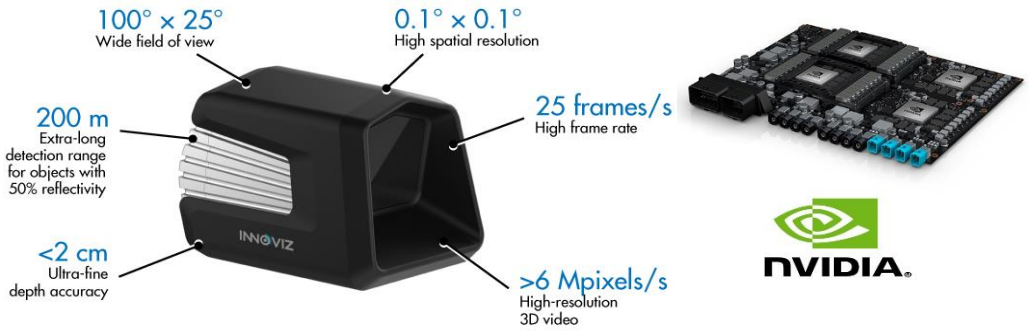
Simulation : WAYMO **:** 4,35 milliards de km

NVIDIA : Drive constellation ⟶ réalisme, météo, imprévu

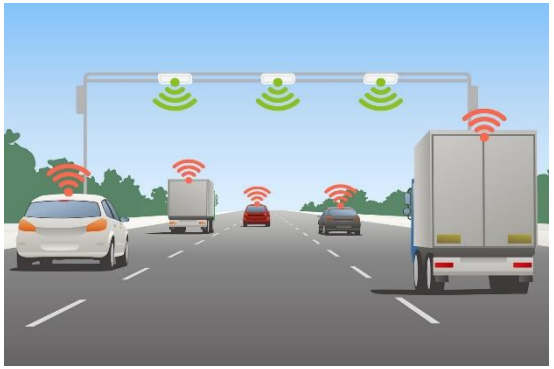
R&D : ENABLE-S3

Collecte : TESLA ‘shadow mode’, Cartes 3d HD enrichies

Evolutions matérielle

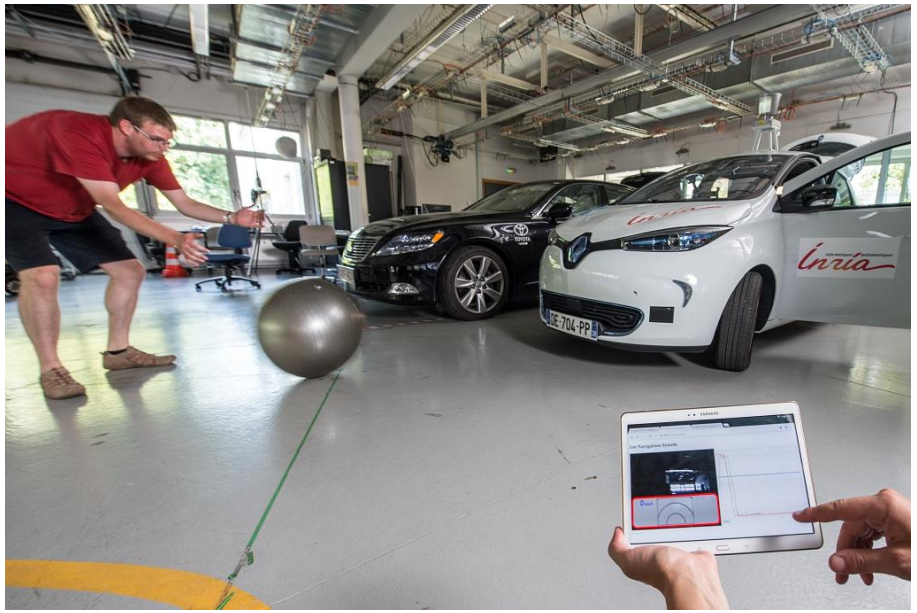


Communications V2X



Inria inventeurs du monde numérique

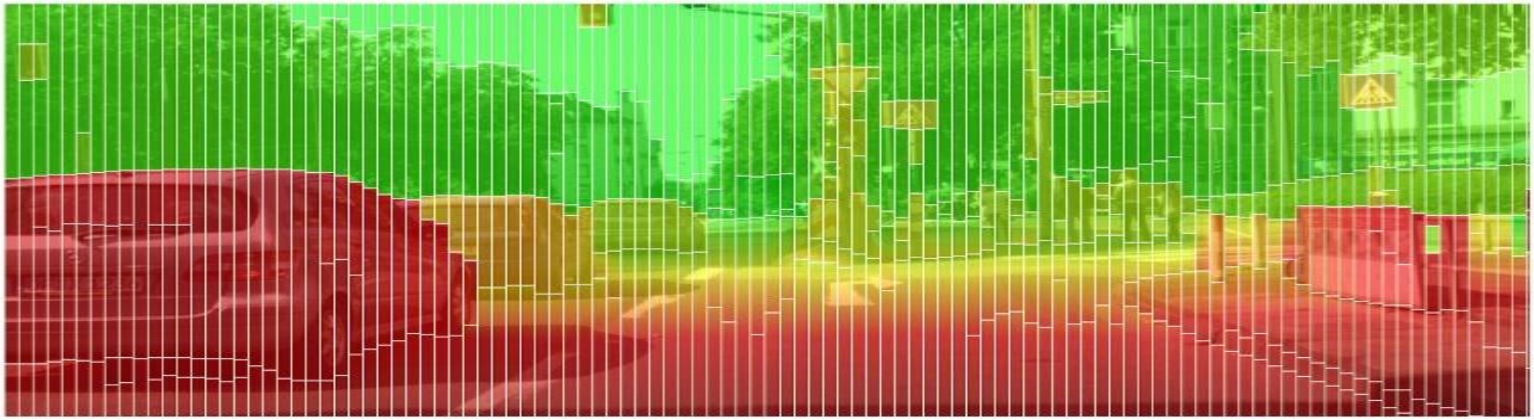
Merci de votre attention !



inventeurs du monde numérique

Stixels

Semantic representation, where Stixel colors encode semantic classes following [20].



Depth representation, where Stixel colors encode disparities from close (red) to far (green).