

Detection and Tracking of Moving Objects

Olivier Aycard

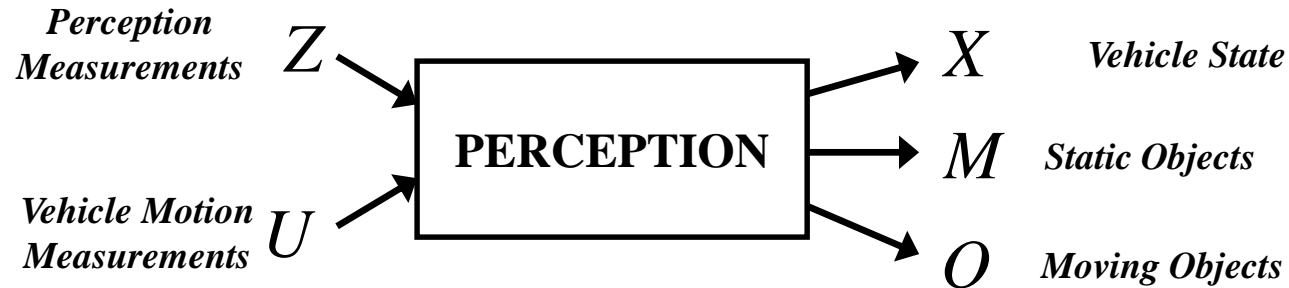
Associate Professor at University of Grenoble1

Laboratoire d'Informatique de Grenoble

<http://membres-liglab.imag.fr/aycard/>

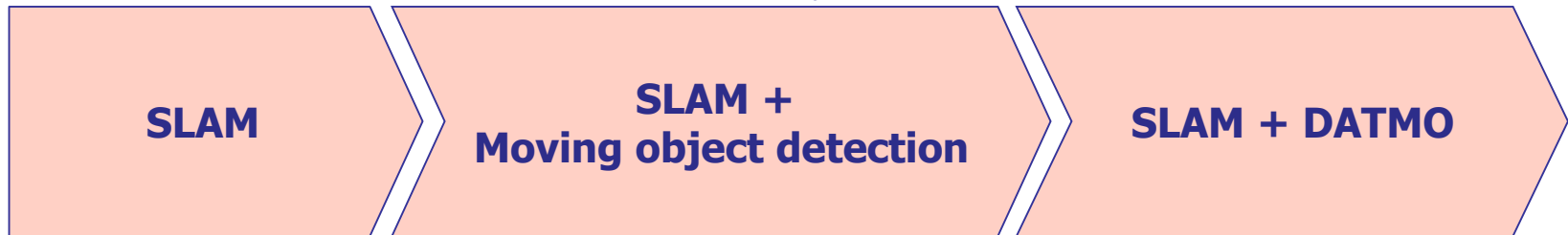


Problem statement



Static environments

Dynamic environments



$$P(\underline{X, M} \mid Z, U)$$

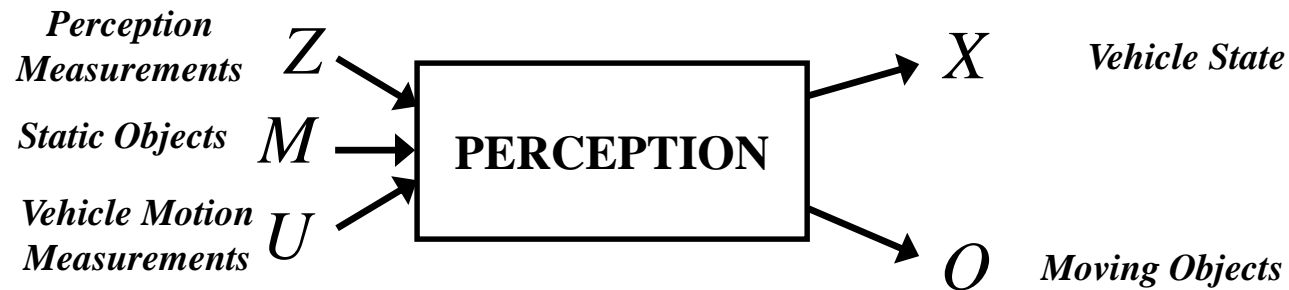
$$\begin{cases} \underline{Z = Z^{(s)} + Z^{(d)}} \\ P(X, M \mid Z^{(s)}, U) \end{cases}$$

$$P(\underline{X, M, O} \mid Z, U)$$

$$\begin{cases} P(X, M \mid Z^{(s)}, U) \\ P(O \mid Z^{(d)}) \end{cases}$$

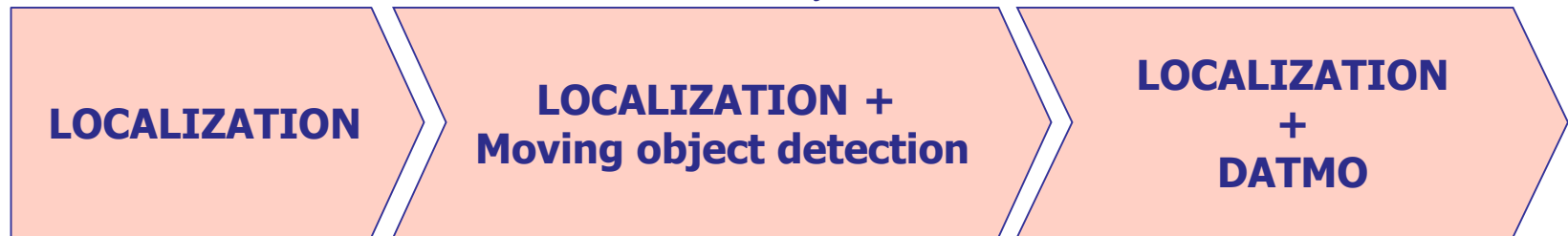
Problem statement

If the map is known: a localization problem



Static environments

Dynamic environments



$$\underline{P(X | Z, U, M)}$$

$$\begin{cases} \underline{Z = Z^{(s)} + Z^{(d)}} \\ P(X | Z^{(s)}, U, M) \end{cases}$$

$$\underline{P(X, O | Z, U, M)}$$

$$\begin{cases} P(X | Z^{(s)}, U, M) \\ P(O | Z^{(d)}) \end{cases}$$

The 2 main perception problems

Localization problem: we want to find the position/state of a mobile robot in its environment

- A map of the environment is given
- The mobile robot moves in its environment
- The mobile robot perceives its environment

$$P(X | Z^{(s)}, U, M)$$

Tracking problem: we want to find the position/state of a moving object (or several moving objects) in its environment

- A map of the environment is not needed
- We estimate the motion of the moving objects
- We perceive the moving objects

$$\begin{aligned} P(O | Z^{(d)}) \\ = P(O | Z^{(d)}, (U), M) \end{aligned}$$

Very similar practical problems

Similar theoretical problems

=> same tool/technique to solve them: Bayesian filters

Detection of Moving Objects

Goal:

- Detect moving objects using a lidar

Problem:

- Distinguish changes in the lidar scanner due to motion

Assumptions:

- Fixed (or known position) of the mobile robot

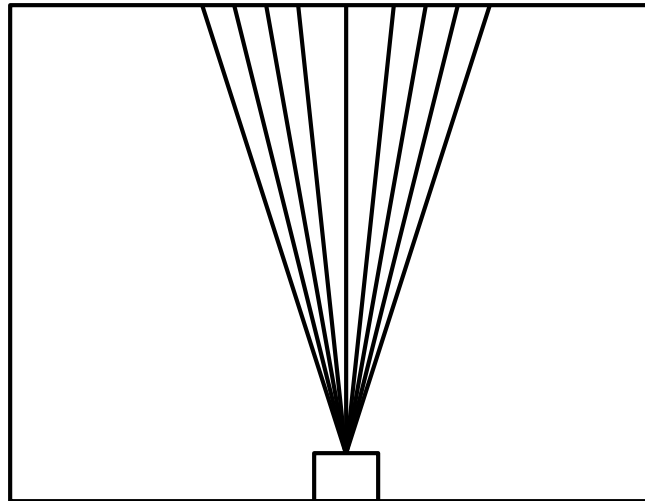
Advantages:

- No a priori knowledge on object dynamic
- No a priori knowledge on object nature

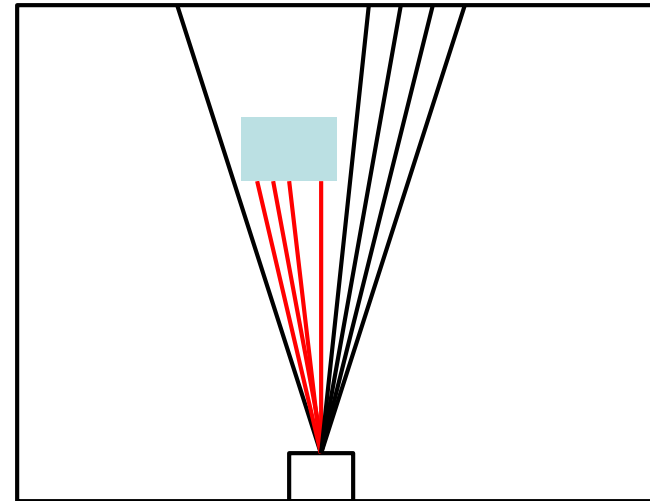
Detection of Moving Objects(1/3)

A very simple idea (but it works):

- Difference for each beam b between the current laser scanner acquisition and the initial acquisition
 - $d_1(b) - d_T(b) > \text{threshold}$ for a beam b
- ⇒ If yes: a moving object should be present at beam b



Initial time $t = 1$



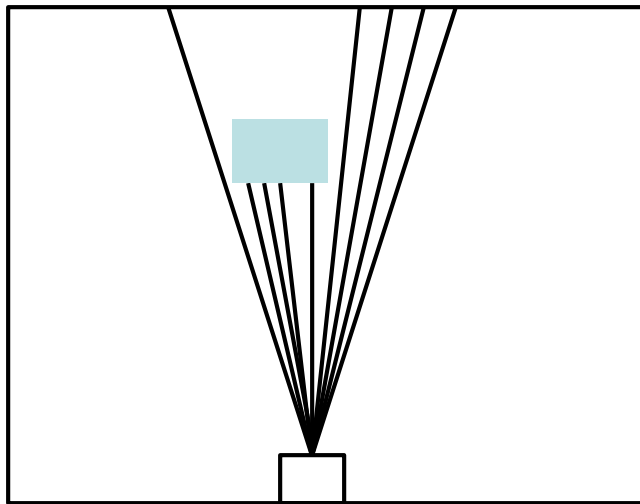
current time $t = T$

Need of a background image (ie, $d_1(b)$ for each b)

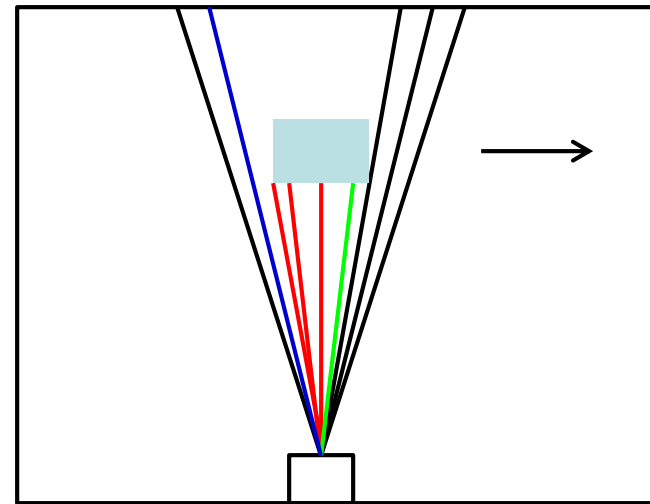
Detection of Moving Objects(2/3)

An other idea:

- Difference for each beam b between the current laser scanner acquisition and the previous one
 - $d_T(b) - d_{T-1}(b) > \text{threshold}$ for one or several beam(s) b
 - $d_T(b+db) = d_{T-1}(b+db)$ for one or several beam(s) b
 - $d_{T-1}(b+db') - d_T(b+db') > \text{threshold}$ for one or several beam(s) b'
- ⇒ If yes: a moving object should be moving from left to right



previous time

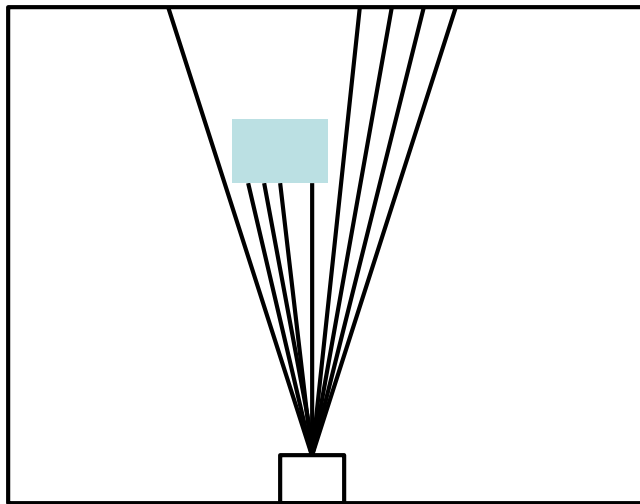


current time

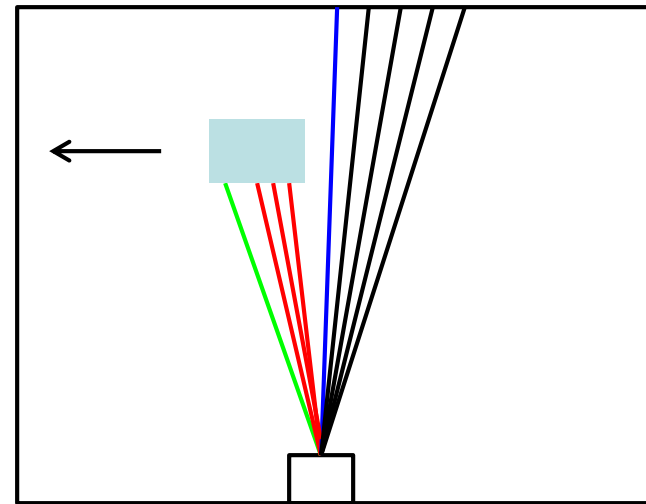
Detection of Moving Objects(3/3)

An other idea:

- Difference for each beam b between the current laser scanner acquisition and the previous one
 - $d_T(b+db') - d_{T-1}(b+db') > \text{threshold}$ for one or several beam(s) b'
 - $d_T(b+db) = d_{T-1}(b+db)$ for one or several beam(s) b
 - $d_{T-1}(b) - d_T(b) > \text{threshold}$ for one or several beam(s) b
- If yes: a moving object should be moving from right to left



previous time

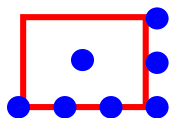


current time

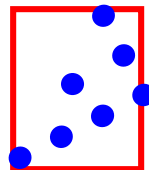
Find the center of gravity of a moving object

To know the position of a detected object, we have to find its center of gravity:

- Find the 2D bounding box surrounding a moving object
- ⇒ Find the minimum and maximum values on each axis for a set of moving points



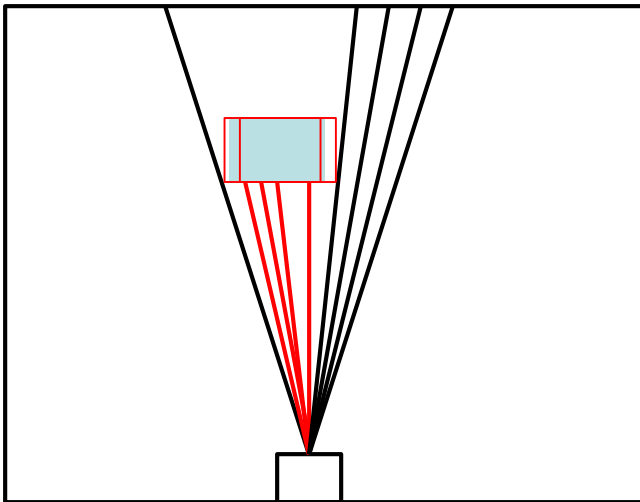
Good value



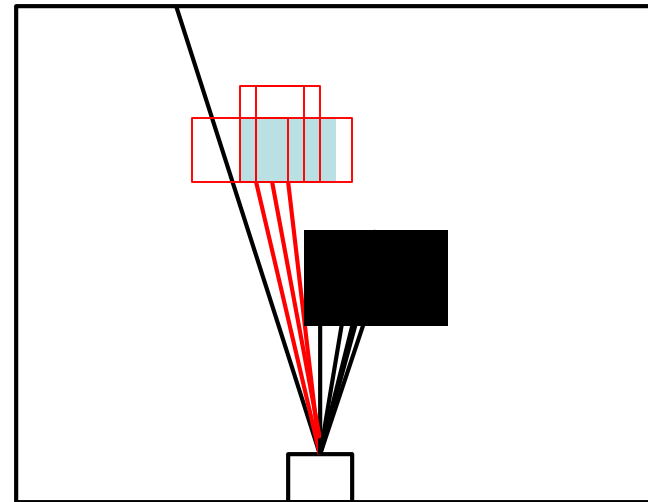
bad value

It is not always possible to know the position of a moving object

- We are not always able to know the size of the object



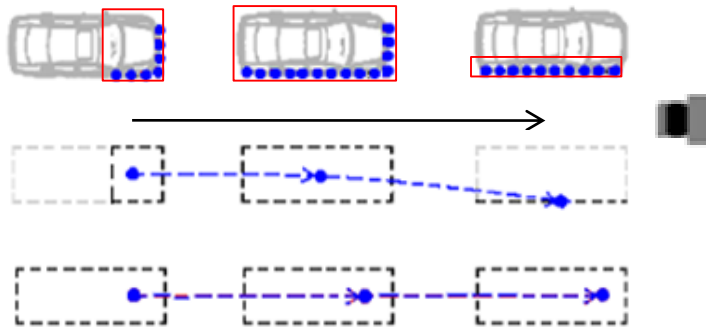
Extremities of the object
are not visible
Many hypothesis could
correspond to the real object
position




Object is partially occluded
Still more hypothesis could
correspond to the real object
position

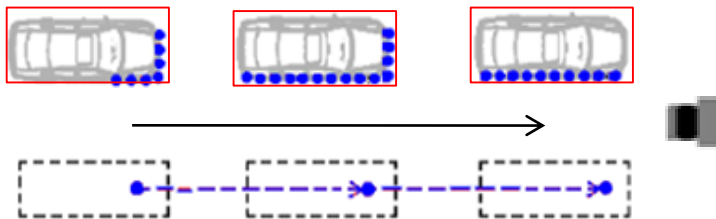
Use a model of detected objects

Objects are represented by groups of points [Burlet'07]

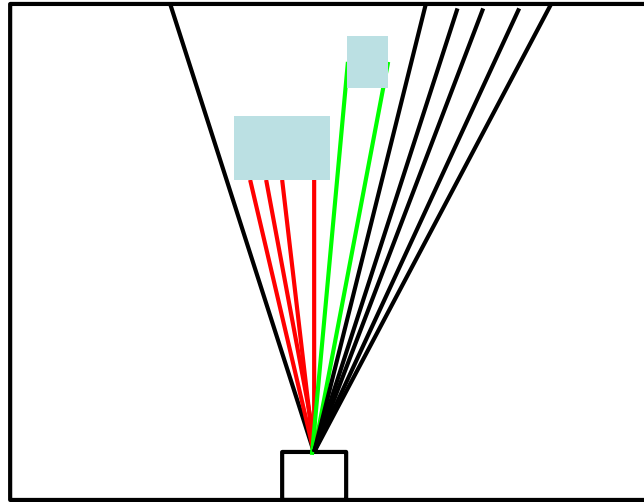
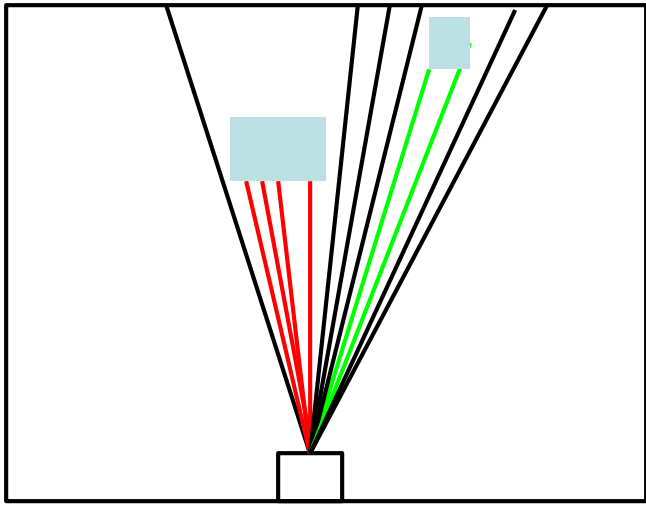


Detecting groups of points leads to incorrect results

- ⇒ Model-based approach can overcome these problems [Dung'09]
- ⇒ Suppose that moving objects are a specific model of car.
- ⇒ We know the size of this specific model of car. 



Cluster of raw data to form objects (1/2)



Cluster of raw data to form objects (2/2)

// Initialization of the first cluster

Initialize the last beam with the first beam that is considered as dynamic

Create a first cluster with the last beam

// Loop over all the beams that are considered as dynamic

For each beam that is considered as dynamic

If the distance between the current beam and the last beam is lower than a given threshold

Then add the current beam to the current cluster

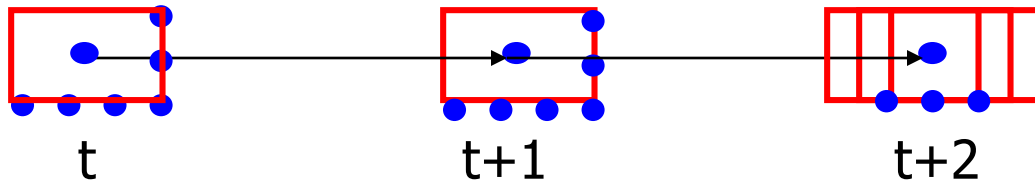
Else create a new cluster with the last beam

last beam = current beam

End for

Tracking of Moving Objects

To have a good estimation of position even if detection is not good enough, we use historic of detection and estimation of motion to track the position of a moving object



1. Estimation of motion between t and $t+1$
2. Prediction of position at $t+2$ using estimation of motion between t and $t+1$

**It works if estimation of motion is perfect !!!
Otherwise, we have to combine both uncertainty
(motion and detection) to find the best
estimation of position of moving objects**

Kalman Filter

- Kalman Filter is an implementation of Bayesian Filters
- State space is represented by a gaussian distribution
- S_t : state at time t (S_t is a gaussian distribution represented by $N(\mu_t, \Sigma_t)$)
- O_t : observation at time t
- A_t : actions at time t
- Hypothesis :
 - Order 1 Markov model
 - $P(S_t / S_{t-1}, A_t)$: S_t is a linear combination of S_{t-1} and A_t . The noise associated to this model is gaussian.
 - Sensor model
 - $P(O_t / S_t)$: O_t is a linear combination of S_t . The noise associated to this model is gaussian.
- Goal : compute *prior* distribution $P(S_T | O_{0:T}, A_{1:T})$

Kalman filter: assumptions(1/2)

- S_t is a gaussian distribution represented by $N(\mu_t, \Sigma_t)$
- Dynamic model is linear and associated noise is gaussian
 - $S_t = S_{t-1} + BA_t + \varepsilon_t$
 - S_t is a n vector
 - A_t is a m vector
 - B is $n \times m$ matrix: transformation from action space to state space
 - ε_t is a gaussian distribution represented by $N(0, R_T)$ modeling the noise associated with the dynamic model

$$P(S_T | S_{T-1}, A_T) = \frac{1}{(2\pi)^{\frac{N}{2}} \sqrt{\det(R_T)}} e^{-\frac{1}{2}((S_T - S_{T-1} - BA_T)^T R_T^{-1} (S_T - S_{T-1} - BA_T))}$$

Kalman filter: assumptions(2/2)

- Sensor model is linear and associated noise is gaussian
 - $O_t = CS_t + \delta_t$
 - O_t is a k vector
 - C is a $k \times n$ matrix: transformation from state space to observation space
 - δ_t is a gaussian distribution represented by $N(0, Q_t)$ modeling the noise associated with the observation model

$$P(O_T | S_T) = \frac{1}{(2\pi)^{\frac{N}{2}} \sqrt{\det(Q_T)}} e^{-\frac{1}{2}((O_T - CS_T)^T Q_T^{-1} (O_T - CS_T))}$$

- **S_{t+1} is a gaussian distribution represented by $N(\mu_{t+1}, \Sigma_{t+1})$**
 - Only mean and covariance needed;
 - Exact inference using simple linear equations

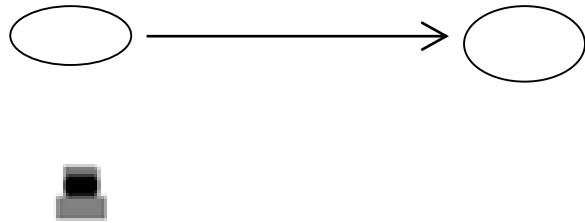
Kalman filter: equations

$$\left. \begin{aligned} \mu_{t+1} &= \mu_t + BA_t \\ \Sigma_{t+1} &= \Sigma_t + R_t \end{aligned} \right\} \text{Prediction}$$

$$K_t = \Sigma_{t+1} C^T (C \Sigma_{t+1} C^T + Q_t)^{-1} \text{ Kalman gain}$$

$$\left. \begin{aligned} \mu_{t+1} &= \mu_{t+1} + K_t (O_t - C_t \mu_{t+1}) \\ \Sigma_{t+1} &= (I - K_t C_t) \Sigma_{t+1} \end{aligned} \right\} \text{Estimation}$$

Tracking of a moving object (1/6)



- A moving object is moving in a direction perpendicular to the laser scanner
- It is initially located at 1 meter on the x-axis;
- We don't perceive the complete object (uncertainty on detection);
- We estimate it is moving on the x-axis with a speed of about 3 meters per second (uncertainty on motion).

Kalman filter: equations and parameters for our tracking problem

μ_t is the position of the moving object on x-axis (ie, state space)

A_t is the estimation of motion of the moving object on x-axis (ie, action space)

R_t uncertainty associated to motion is small (ie, 2)

O_t is the detection of the moving object on x-axis (ie, observation space)

Q_t uncertainty associated to detection is small (ie, 2)

B and C are identity because the state space, the action space and the observation space are the same space (ie, x-axis)

Kalman filter: equations

$$\left. \begin{aligned} \mu_{t+1} &= \mu_t + A_t \\ \Sigma_{t+1} &= \Sigma_t + R_t \end{aligned} \right\} \text{Prediction}$$

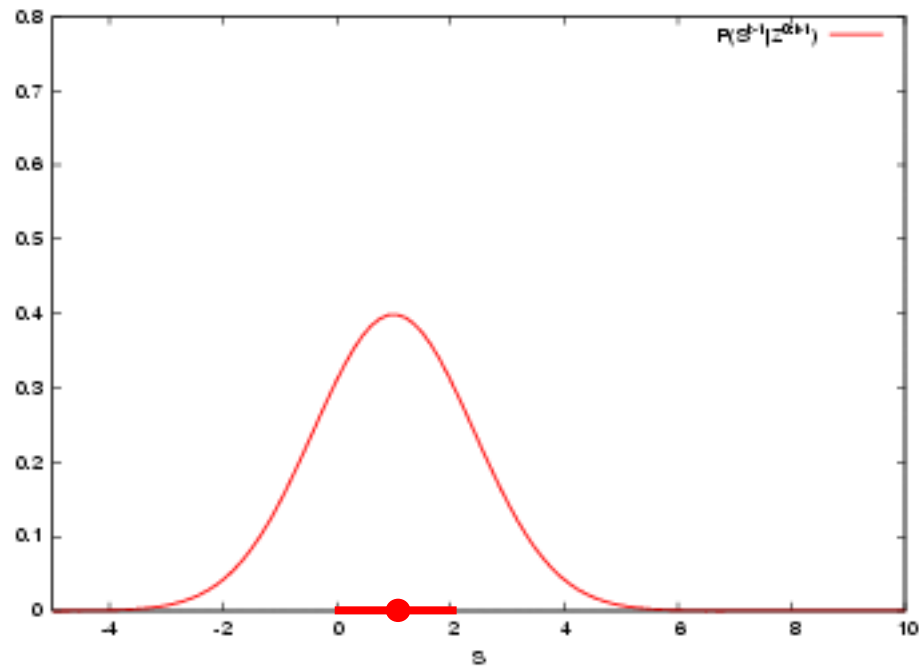
$$K_t = \Sigma_{t+1} / (\Sigma_{t+1} + Q_t) \text{ Kalman gain}$$

$$\left. \begin{aligned} \mu_{t+1} &= \mu_{t+1} + K_t(O_t - \mu_{t+1}) \\ \Sigma_{t+1} &= (1 - K_t)\Sigma_{t+1} \end{aligned} \right\} \text{Estimation}$$

Example: initial stage(2/6)

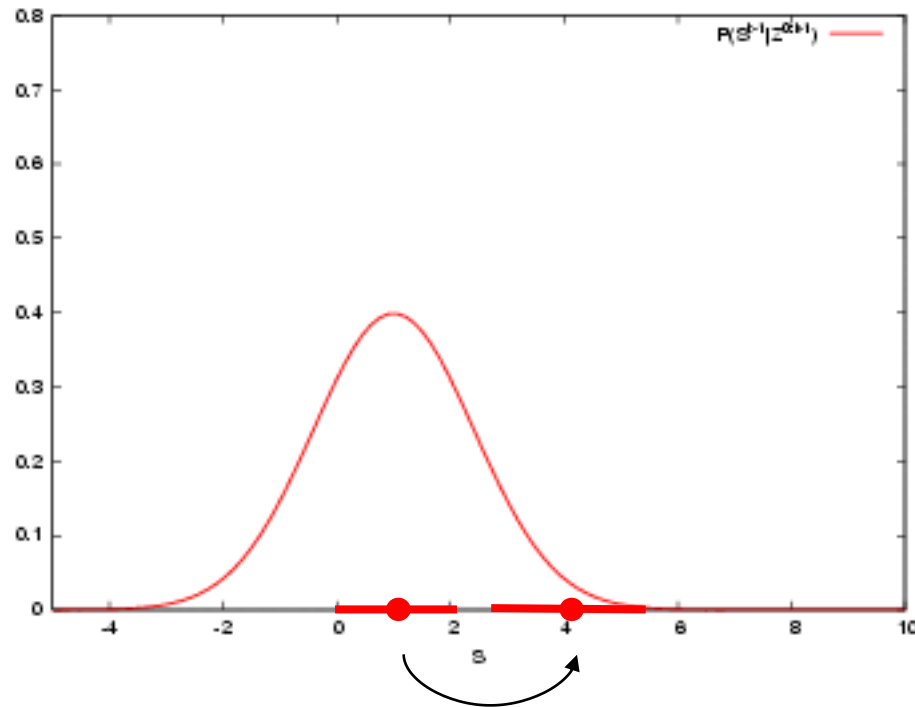
$$\mu_0 = 1$$

$$\Sigma_0 = 2$$



Example (3/6): prediction stage

$$\mu_1 = \mu_0 + A_1 = 1 + 3 = 4$$

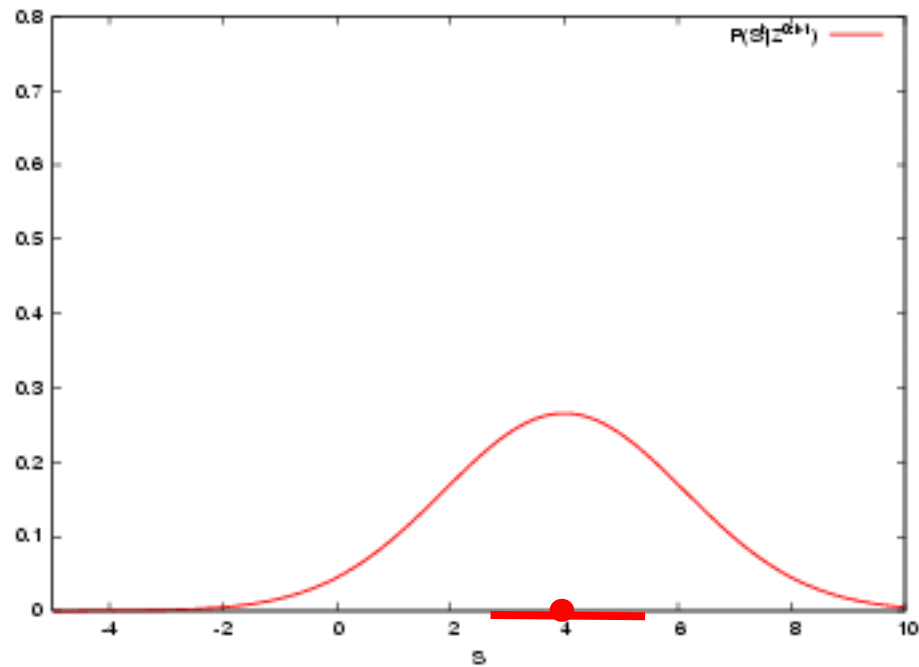


Dynamic model

Example(4/6): prediction stage

$$\mu_1 = 4$$

$$\Sigma_1 = \Sigma_0 + R_1 = 2 + 2 = 4$$

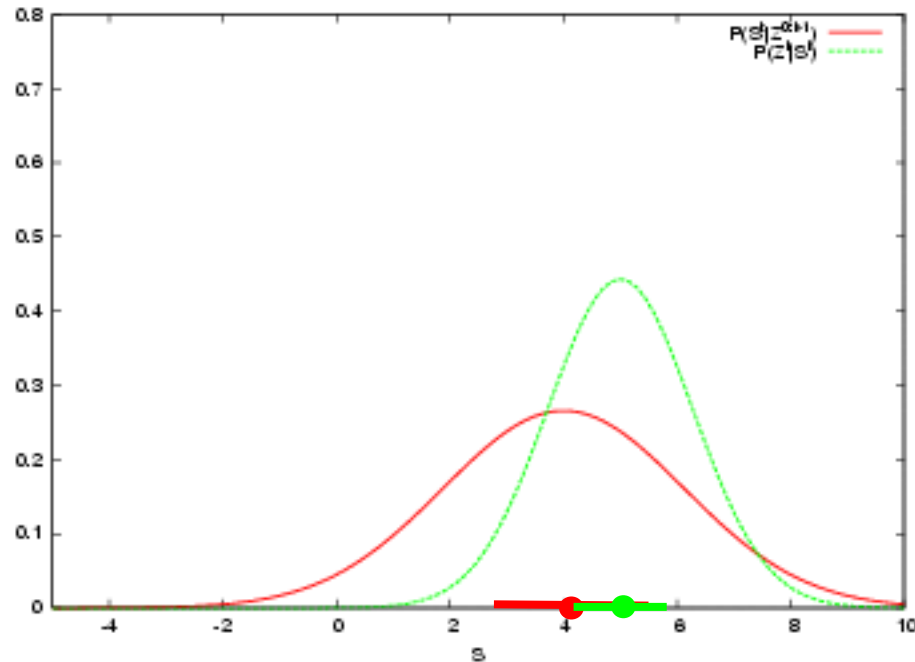


$$P(S_1|A_1)$$

Example (5/6): estimation stage

$$O_1 = 5$$

$$K_1 = \Sigma_1 / (\Sigma_1 + Q_1) = 4 / (4 + 2) = 2/3$$

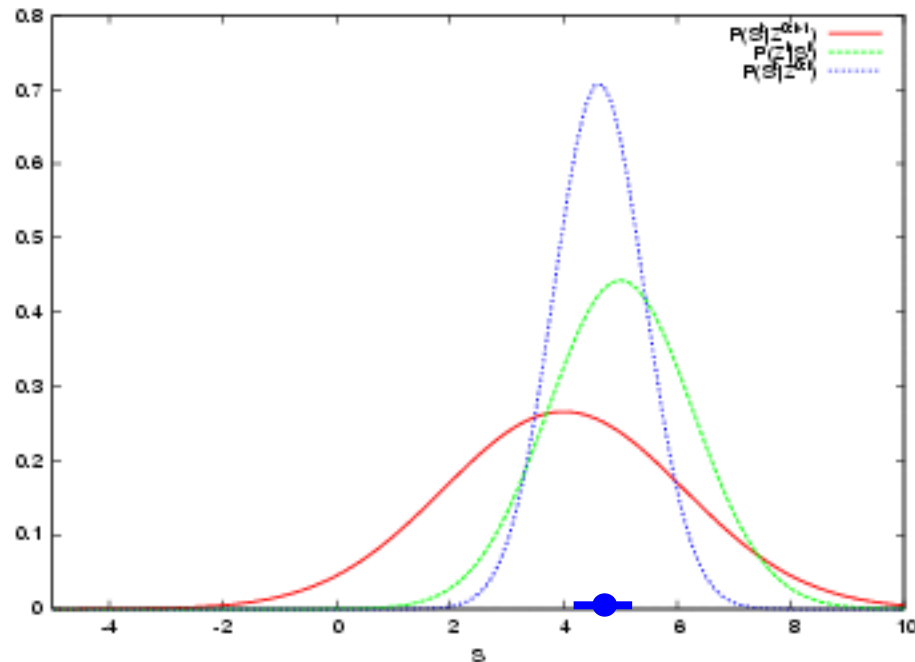


Sensor observation at time t

Example (6/6): estimation stage

$$\mu_1 = \mu_1 + K_1(O_1 - \mu_1) = 4 + 2/3 \times (5 - 4) = 4 + 2/3 \times 1 = 14/3$$

$$\Sigma_1 = (1 - K_1)\Sigma_1 = (1 - 2/3) \times 4 = 4/3$$

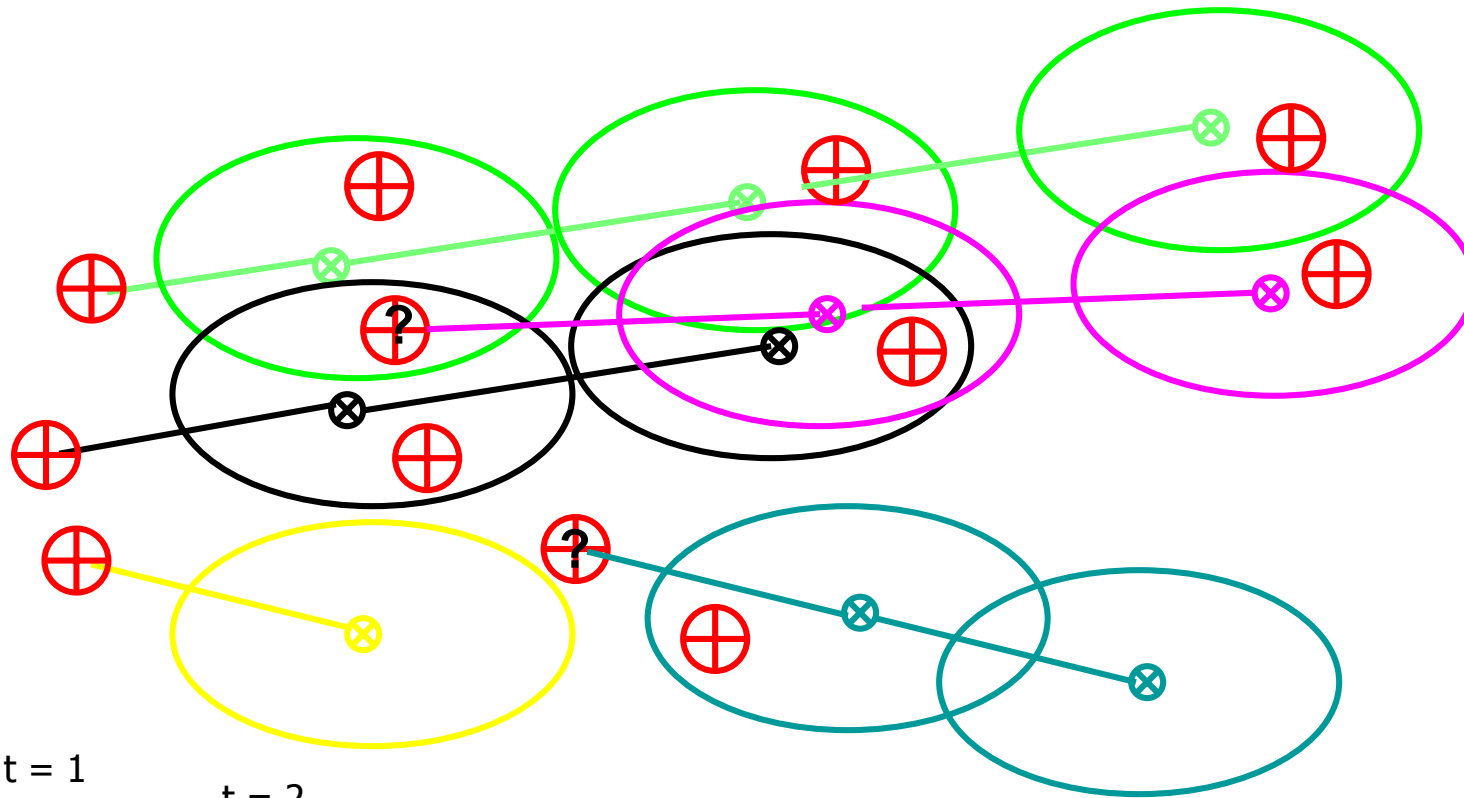


Posterior pdf at time t
 $P(S_1|A_1, O_1)$

Conclusion

- KF limited to *linear* models (sensor and dynamic)
- KF limited to gaussian noise
- KF can only represent monomodale distributions: only tracking
- EKF linearizes the estimation around the current estimate
 - Local linearization using Taylor expansion
 - Similar to KF (ie, gaussian distribution for the state space)
- KF and EKF popular and successful
- Derived models:
 - Unscented Kalman Filter (dynamic model non linear);
 - Multi Hypothesis Kalman Filter.

Multi objects detection and tracking



t = 1
3 possible
tracks
have to be
confirmed

t = 2
Green and black
tracks
are confirmed.
Yellow is a false alarm.
Two new observations
could correspond to
new tracks.

t = 3
Green, purple and blue
tracks are confirmed.
Black track has left the
field of view.

t = 4
Green and purple
tracks
are confirmed.
Blue track has left the
field of view.

Multi objects detection and tracking

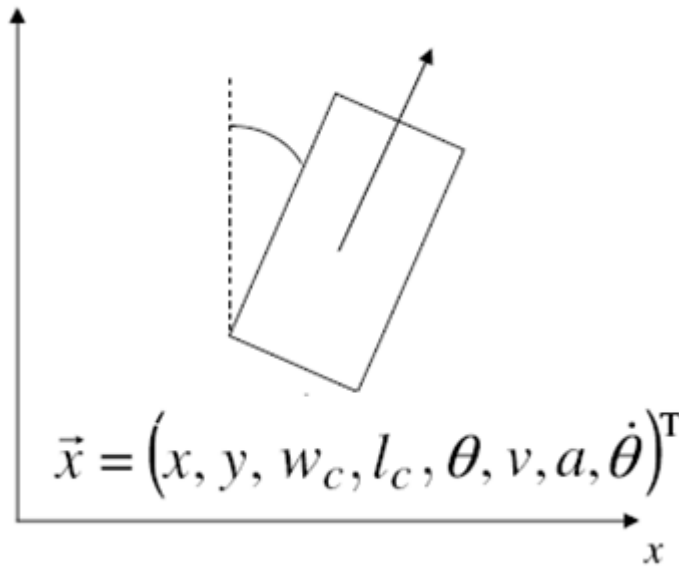
- The number of moving objects is unknown
- Even if it is known at a given time, it could change in the future: moving objects enters and leaves the field of view of the sensor
- The sensors are noisy:
 - False alarm: some noise on a sensor could be interpreted as a moving object
 - Miss detection: sometimes a moving object could not be detected by a sensor (a moving object could hide an other obe)
- We never know how to associate an observation to a track: the number of detections and the number of tracks are different

TO DO

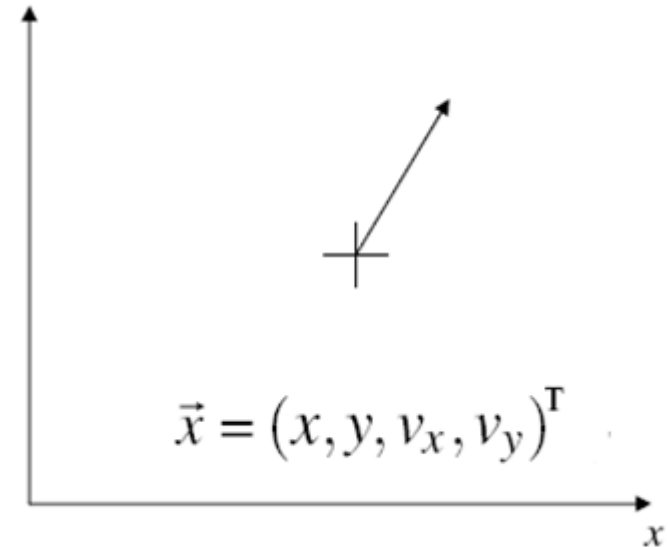
- Rajouter un clustering pour enlever tous les bruits parasites et le détailler dans le cours
- Détailler le kf sur plusieurs itérations et changer les valeurs des paramètres: action parfaite et observation parfaite
- Faire le lien avec un tp de détection: il faudrait un carton pour avoir un objet propre
- On peut finir avec du MOT
- Even if it is known at a given time, it could change in the future: moving objects enters and leaves the field of view of the sensor
- The sensors are noisy:
 - False alarm: some noise on a sensor could be interpreted as a moving object
 - Miss detection: sometimes a moving object could not be

Object models

- Four classes of moving objects:
 - Bus, car, bike (bicycles, motorcycles), pedestrian



- Box-model of fixed-size
 - bus, car, bike

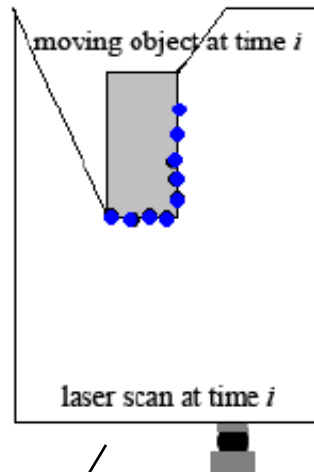


- Point-model:
 - pedestrian

DATMO – Likelihood probability $P(Z | \omega)$

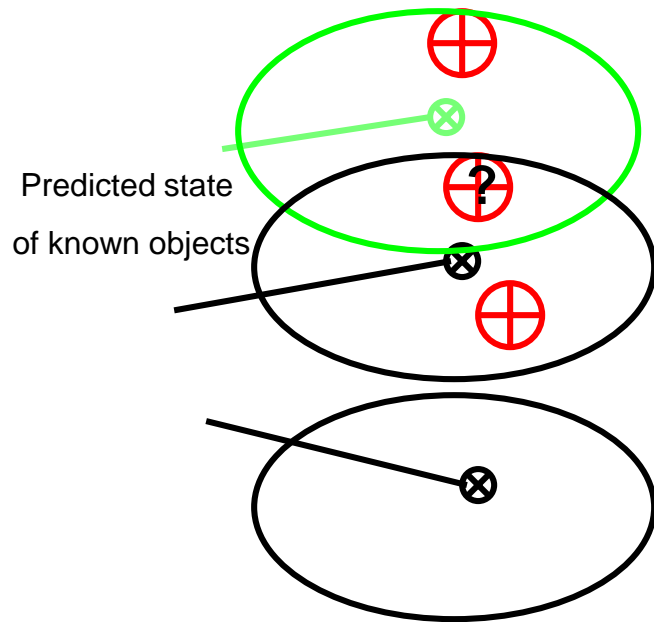
- Identify laser rays caused by moving objects in ω , ω considered as a partition of observation Z into: $Z_t = Z_t^{(d)} + Z_t^{(s)}$

$$p(Z|\omega) = \prod_i P_{M_1}(Z_i^{(d)} | \omega_i)$$



$$P_{M_1}(Z_i^{(d)} | \omega_i) = \prod_{n=1}^N P(z_i^n | \omega_i)$$

DATMO – example



New observations

Gating

Association refinement

- nearest neighbour
- probabilistic techniques : JPDA, MHT

Object refinement

- Confirmation
- Destruction
- Creation

Difficulties:

- number of objects is unknown;
- objects could be occluded;
- sensor faults (missed detection, ghosts or false positive);

Detection of Moving Objects

Detection:

Parler des 2 techniques de détection :

- nécessité d'une image de fond et pas d'estimation de la direction du mouvement
- différence entre 2: pas d'image de fond et estimation de la direction du mouvement

Clustering pour trouver les objets et éliminer le bruit

Modéliser l'objet : centre de gravité

TP avec un carton pour que ce soit simple

Partie où on ne voit pas les objets, lien avec tracking

Multi objets et association

Tracking of Moving Objects

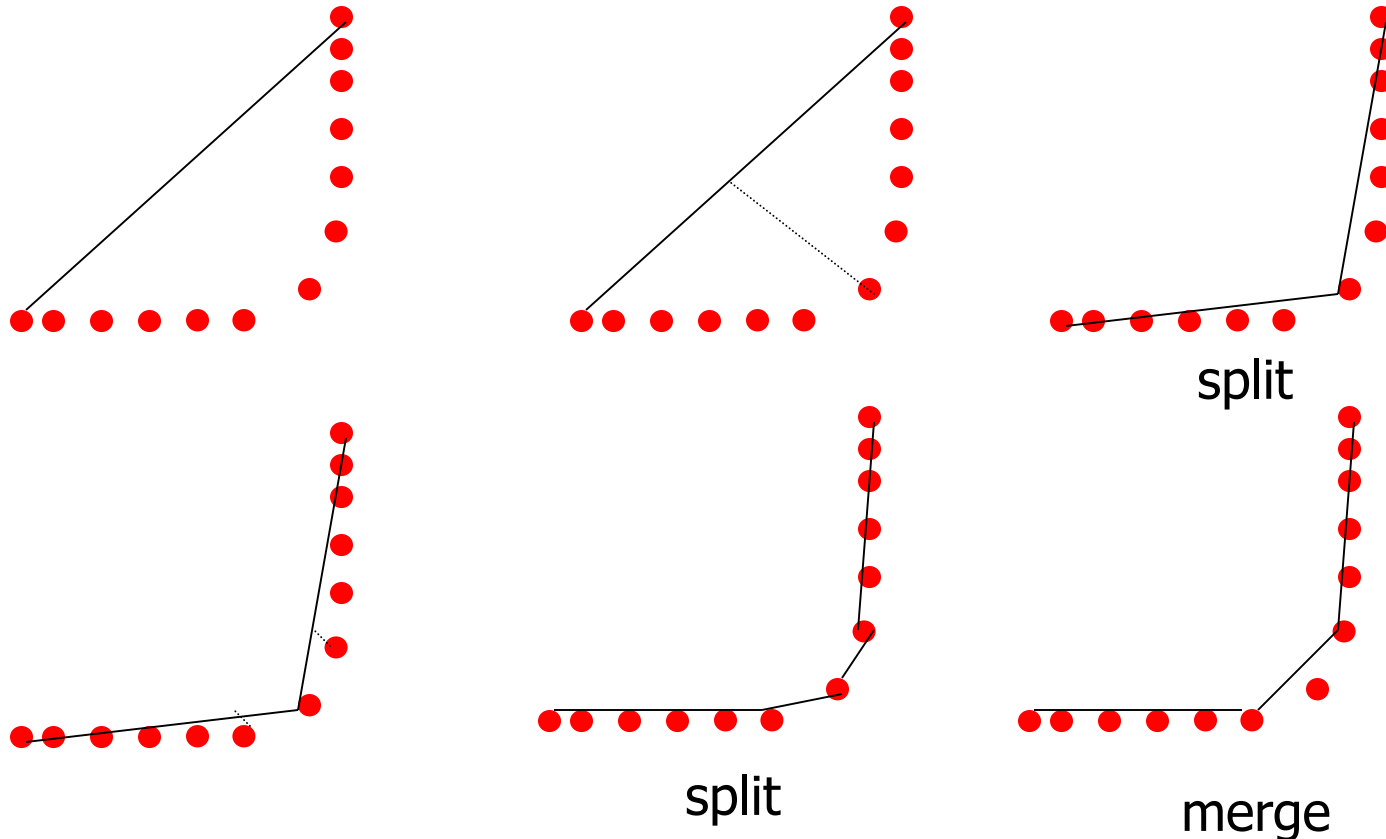
Split & merge algorithm:

- **Split**
 - Obtain the line passing by the two extreme points
 - Find the most distant point to the line
 - If distance $>$ threshold, split
 - Repeat with the left and right point sets
- **Merge**
 - If two consecutive segments are close/collinear enough,
 - Obtain the common line and find the most distant point
 - If distance \leq threshold, merge both segments

Tracking of Moving Objects

How to extract lines from a set of points ?

Split & merge algorithm:



Tracking of Moving Objects

To track a moving object, we need to define the point to track: “point-based tracking”

1. track the center of gravity of extracted lines
 - Extract lines for the set of moving points
 - Find the center of gravity



Find the center of gravity of a moving object

Extraction de lignes

Expliquer que quand l'objet n'est pas complètement visible, il y a une incertitude sur le positionnement

on combine cette incertitude avec l'incertitude sur le mouvement pour obtenir le meilleur positionnement possible

lien avec le filtrage

compensation des 2 incertitude

problème lorsqu'on n'a pas de détection d'un objet

Cas multi-objets: clustering des objets et data association