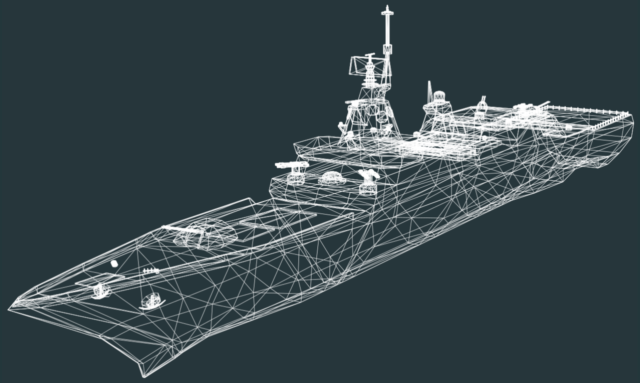


Bayesian methods for Extended Object Tracking

Matteo Tesori, PhD

matteotesori@gmail.com

NATO STO CMRE, La Spezia, January 1st 2026

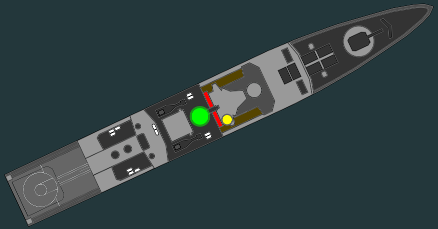
Outline

Introduction	whoami, problem definition, state of the art and motivation
Part 1	tracking for maneuvering objects
Part 2	tracking for elliptical objects
Part 3	tracking for general objects
Conclusions	summary and future research directions

Introduction

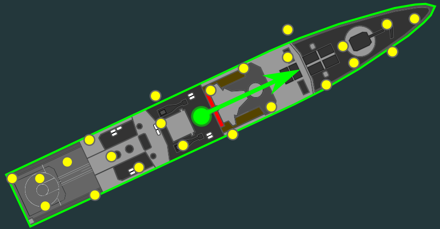
Whomai

Problem definition



Point Object Tracking

single point ⟶ position



Extended Object Tracking

point cloud ⟶ position, orientation, shape

Extended Object Tracking (EOT) problem

Given the time sequence of point clouds $\mathcal{Y}_1, \ldots, \mathcal{Y}_k$, estimate in a Bayesian fashion the state x_k of the extended object (including position, orientation and shape).

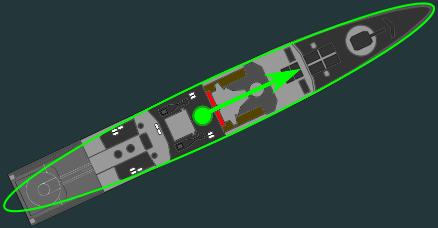
Random Matrix Model [Koch]

characteristic equation

$$z(s,\theta) \triangleq p + s\,U(h)\begin{bmatrix} a\,\cos\theta \\ b\,\sin\theta \end{bmatrix}$$

parameters

position	$p \in \mathbb{R}^2$
heading	$h \in [-\pi, \pi)$
semi-length	$a \in \mathbb{R}_{>0}$
semi-width	$b \in \mathbb{R}_{>0}$



Filters

Gaussian-Inverse-Wishart [Koch]

MEM-EKF* [Baum]

State of the art

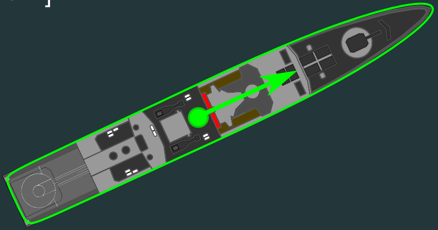
Random Hypersuperface Model [Baum]

characteristic equation

$$z(s,\theta) \triangleq p + s\,U(h)\,\rho(\theta)$$

parameters

position	$p \in \mathbb{R}^2$
heading	$h \in [-\pi, \pi)$
radius	$\rho(\cdot) \in \mathcal{C}^0([0, 2\pi])$



Filters

Fourier UKF [Baum]

Radial Gaussian Processes [Wahlstrom]

Motivation

- **Maneuvering objects:** take advantage of the heading information to improve turning rate estimation in coordinated turn models (and their generalizations).
  PRO: discard interacting multiple models in prediction.
  CON: fragile to measurement noise.

Motivation

- **Efficient statistics:** rather than process each point in $\mathcal{Y} = \{y_1, \ldots, y_m\}$, process

$$\overline{y} \triangleq \frac{1}{m} \sum_{j=1}^{m} y^{(j)}$$

sample mean

$$\overline{Y} \triangleq \frac{1}{m-1} \sum_{j=1}^{m} \left(y^{(j)} - \overline{y}\right) \left(y^{(j)} - \overline{y}\right)'$$

sample covariance

to infer the object's position p, heading h, length $2\ell_1$, width $2\ell_2$.

PRO: cheap computational cost.

CON: loss of information.

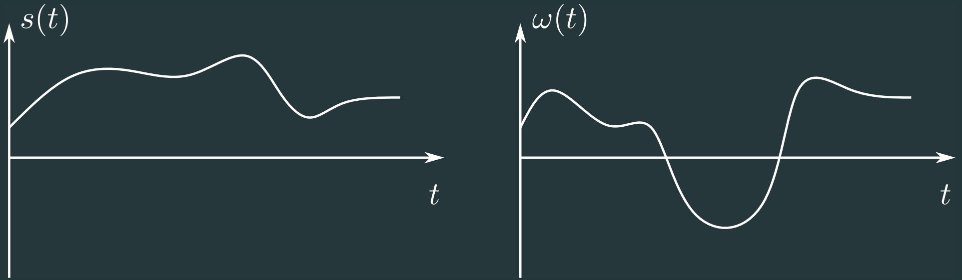
Motivation

- **Shape classification:** cast shape estimation as a classification problem over a known shape family (**shape library**).
  PRO: arbitrarily complex shapes can be recognized.
  PRO: robustness to occlusion.
  CON: only known shapes can be handled.

Tracking for maneuvering objects

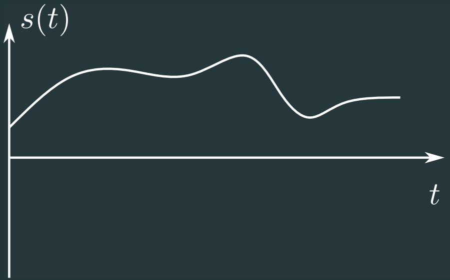
Tracking for maneuvering objects

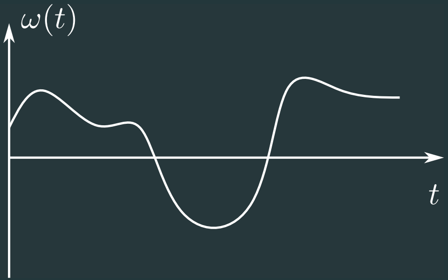
An object is **maneuvering** iff its speed $s(t)$ and/or turning rate $\omega(t)$ vary in time.



Tracking for maneuvering objects

An object is **maneuvering** iff its speed $s(t)$ and/or turning rate $\omega(t)$ vary in time.

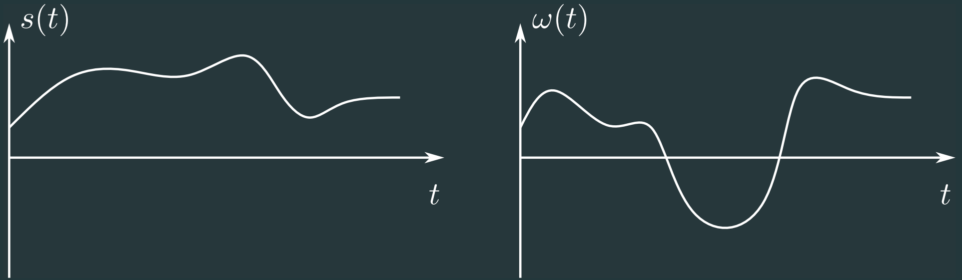




(1) such variables are useful to improve position and heading predictions.

Tracking for maneuvering objects

An object is **maneuvering** iff its speed $s(t)$ and/or turning rate $\omega(t)$ vary in time.

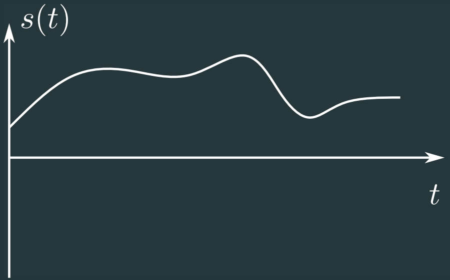


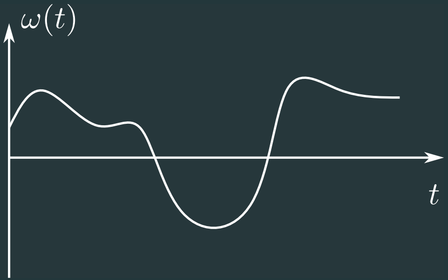
(1) such variables are useful to improve position and heading predictions.

(2) in EOT, we can "directly observe" position and heading from data.

Tracking for maneuvering objects

An object is **maneuvering** iff its speed $s(t)$ and/or turning rate $\omega(t)$ vary in time.

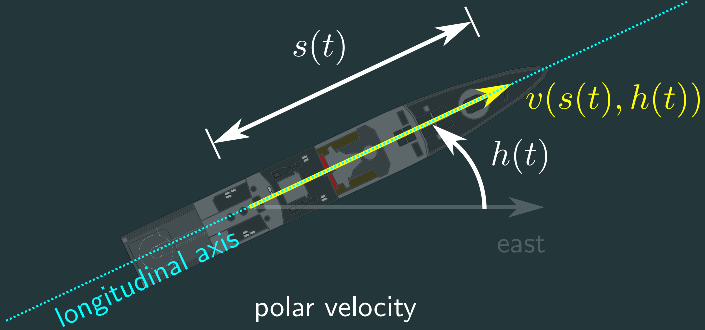




(1) such variables are useful to improve position and heading predictions.

(2) in EOT, we can "directly observe" position and heading from data.

IDEA: define a prediction model to estimate s, ω and their derivatives



polar velocity

$$v(s, h) \triangleq s \begin{bmatrix} \cos h \\ \sin h \end{bmatrix}$$

motion dynamics

(unicycle)

$$\dot{p}(t) = v(s(t), h(t))$$
$$\dot{h}(t) = \omega(t)$$

input dynamics

$$s^{(\Lambda)}(t) \triangleq 0$$
$$\omega^{(O)}(t) \triangleq 0$$

Tracking for maneuvering objects

Kinematic state

$$x \triangleq \begin{bmatrix} p' & \ell' \end{bmatrix}' \qquad \ell \triangleq \begin{bmatrix} h & s & \dots & s^{(\Lambda-1)} & \omega & \dots & \omega^{(O-1)} \end{bmatrix}'$$

Tracking for maneuvering objects

Kinematic state

$$x \triangleq \begin{bmatrix} p' & \ell' \end{bmatrix}' \qquad \ell \triangleq \begin{bmatrix} h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} \end{bmatrix}'$$

Dynamics discretization

$$\begin{aligned} \dot{p}(t) &= v(\ell(t)) \\ \dot{\ell}(t) &= A\ell(t) \end{aligned} \qquad \Rightarrow \qquad \begin{aligned} p_k &= p_{k-1} + \int_{(k-1)T}^{kT} v(\ell(\tau))\ \mathsf{d}\tau \\ \ell_k &= \exp(AT)\,\ell_{k-1} \end{aligned}$$

Tracking for maneuvering objects

Kinematic state

$$x \triangleq \begin{bmatrix} p' & \ell' \end{bmatrix}' \qquad \ell \triangleq \begin{bmatrix} h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} \end{bmatrix}'$$

Dynamics discretization

$$\begin{aligned} \dot{p}(t) &= v(\ell(t)) \\ \dot{\ell}(t) &= A\ell(t) \end{aligned} \quad \Rightarrow \quad \begin{aligned} p_k &= p_{k-1} + \int_{(k-1)T}^{kT} v(\ell(\tau))\, \mathsf{d}\tau \\ \ell_k &= \exp(AT)\, \ell_{k-1} \end{aligned}$$

$\Lambda : O$ prediction model

$$\begin{aligned} p_k &= p_{k-1} + T\,\frac{v(\ell_{k-1}) + v(\ell_k)}{2} + w_k^p \\ \ell_k &= \exp(AT)\, \ell_{k-1} + w_k^\ell \end{aligned} \qquad w_k \sim \mathcal{N}(0, Q)$$

Tracking for maneuvering objects

$\varLambda : O$ predictor

$$x_{k|k-1} = \overline{f}_{k|k-1}$$
$$P_{k|k-1} = F_{k|k-1} + Q$$

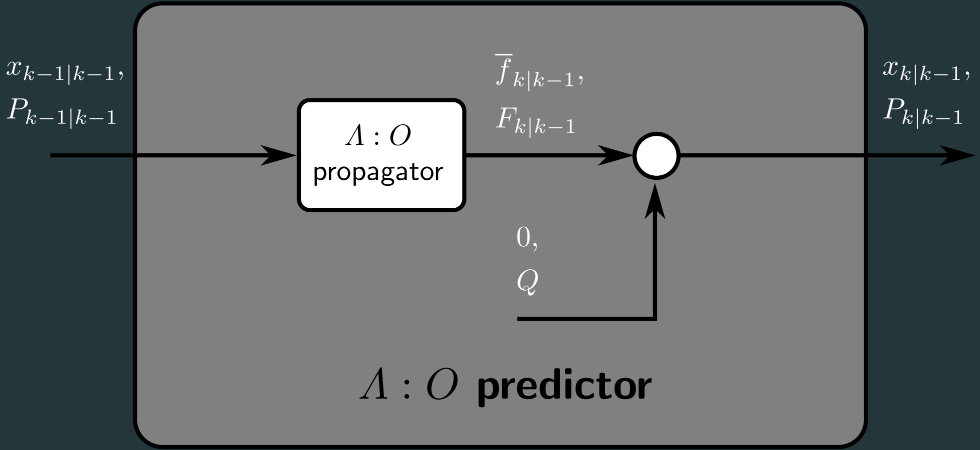
where

$$f(x) \triangleq \left[\; p + T\frac{v(\ell)+v(\exp(AT)\ell)}{2}; \quad \exp(AT)\ell \;\right]$$
$$\overline{f}_{k|k-1} \triangleq \int f(x)\,\mathcal{N}(x; x_{k-1|k-1}, P_{k-1|k-1})\;\mathsf{d}x$$
$$F_{k|k-1} \triangleq \int \left(f(x) - \overline{f}_{k|k-1}\right)\left(f(x) - \overline{f}_{k|k-1}\right)' \mathcal{N}(x; x_{k-1|k-1}, P_{k-1|k-1})\;\mathsf{d}x$$

and the integrals can be computed via:

- linearization (EKF);
- Gaussian quadrature (e.g., UKF, CKF);
- or any other integration method (Grid integration, Importance Sampling, etc...).

Tracking for maneuvering objects



Tracking for elliptical objects

Tracking for elliptical objects

Multiplicative error model (MEM) [Baum]

$$
\begin{aligned}
y &= p + U(h)\,D(e)\,q + v \\
q &\sim \mathcal{N}(0, I/k) \\
v &\sim \mathcal{N}(0, \sigma_v^2 I)
\end{aligned}
\qquad
\begin{aligned}
U(h) &\triangleq \begin{bmatrix} \cos h & -\sin h \\ \sin h & \cos h \end{bmatrix} \\
D(e) &\triangleq \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}
\end{aligned}
$$

Tracking for elliptical objects

Multiplicative error model (MEM) [Baum]

$$y = p + U(h)\, D(e)\, q + v \qquad U(h) \triangleq \begin{bmatrix} \cos h & -\sin h \\ \sin h & \cos h \end{bmatrix}$$

$$q \sim \mathcal{N}(0, I/k)$$

$$v \sim \mathcal{N}(0, \sigma_v^2 I) \qquad D(e) \triangleq \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

Measurement distribution (given p, h, ℓ)

$$y \sim \mathcal{N}\left(p, \quad U(h)\, D\left(\left[\frac{a^2}{k} + \sigma_v^2; \qquad \frac{b^2}{k} + \sigma_v^2\right]\right) U(h)'\right)$$

- $k = 2$ if we model a point cloud distributed over the object contour;
- $k = 4$ if we model a point cloud distributed over the object surface.

Tracking for elliptical objects

Multiplicative error model (MEM) [Baum]

$$y = p + U(h)\, D(e)\, q + v \qquad U(h) \triangleq \begin{bmatrix} \cos h & -\sin h \\ \sin h & \cos h \end{bmatrix}$$

$$q \sim \mathcal{N}(0, I/k)$$

$$v \sim \mathcal{N}(0, \sigma_v^2 I) \qquad D(e) \triangleq \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

Measurement distribution (given p, h, ℓ)

$$y \sim \mathcal{N}\left(p, \quad U(h)\, D\left(\left[\frac{a^2}{k} + \sigma_v^2; \qquad \frac{b^2}{k} + \sigma_v^2\right]\right) U(h)'\right)$$

- $k = 2$ if we model a point cloud distributed over the object contour;
- $k = 4$ if we model a point cloud distributed over the object surface.

IDEA: estimate p, h, ℓ *directly* from $\overline{y} \approx p$, $\overline{Y} \approx \Sigma_y$

QUESTION: what does it mean *directly*?

ANSWER: we define the pseudo-measurement, called *static estimate*, as

$$\mathbb{Y} \triangleq \begin{bmatrix} \hat{p}' & \hat{h} & \hat{e}' \end{bmatrix}'$$

and its covariance

$$\Sigma_{\mathbb{Y}} \triangleq \begin{bmatrix} \Sigma_{\hat{p}} & \Sigma_{\hat{p}\hat{h}} & \Sigma_{\hat{p}\hat{e}} \\ * & \Sigma_{\hat{h}} & \Sigma_{\hat{h}\hat{e}} \\ * & * & \Sigma_{\hat{e}} \end{bmatrix}$$

QUESTION: what does it mean *directly*?

ANSWER: we define the pseudo-measurement, called *static estimate*, as

$$\mathbb{Y} \triangleq \begin{bmatrix} \hat{p}' & \hat{h} & \hat{e}' \end{bmatrix}'$$

and its covariance

$$\Sigma_{\mathbb{Y}} \triangleq \begin{bmatrix} \Sigma_{\hat{p}} & \Sigma_{\hat{p}\hat{h}} & \Sigma_{\hat{p}\hat{e}} \\ * & \Sigma_{\hat{h}} & \Sigma_{\hat{h}\hat{e}} \\ * & * & \Sigma_{\hat{e}} \end{bmatrix}$$

Then we perform a *one-shot* (!!!) Kalman correction based on $\overline{Y}$.

NOTE: mixed terms $\Sigma_{\hat{p}\hat{h}}$, $\Sigma_{\hat{p}\hat{e}}$, $\Sigma_{\hat{h}\hat{e}}$ are neglected for simplicity.

Tracking for elliptical objects

How do we compute the static estimates?

Tracking for elliptical objects

How do we compute the static estimates?

$$\hat{p} \triangleq \overline{y} \qquad \Sigma_{\hat{p}} = \frac{1}{m}\,\overline{Y}$$

Tracking for elliptical objects

How do we compute the static estimates?

$$\hat{p} \triangleq \overline{y} \qquad \Sigma_{\hat{p}} = \frac{1}{m}\,\overline{Y}$$

$$\hat{h} \triangleq \frac{1}{2}\,\mathrm{atan2}\left(2\overline{Y}_{12}, \overline{Y}_1 - \overline{Y}_2\right) \qquad \Sigma_{\hat{h}} = \frac{1}{m-1}\frac{\lambda_1\,\lambda_2}{(\lambda_1-\lambda_2)^2}$$

$$\hat{e} \triangleq \sqrt{k}\begin{bmatrix}\sqrt{\lambda_1-\sigma_v^2}\\ \sqrt{\lambda_2-\sigma_v^2}\end{bmatrix} \qquad \Sigma_{\hat{e}} = \frac{1}{m-1}\,\frac{k}{2}\begin{bmatrix}\frac{\lambda_1^2}{\lambda_1-\sigma_v^2} & \frac{\lambda_1\lambda_2}{2\sqrt{(\lambda_1-\sigma_v^2)(\lambda_2-\sigma_v^2)}}\\ * & \frac{\lambda_2^2}{\lambda_2-\sigma_v^2}\end{bmatrix}$$

where λ_1, λ_2 are the eigenvalues of $\overline{Y} = [\overline{Y}_1, \overline{Y}_{12}; *, \overline{Y}_2]$ and m is the cloud cardinality. $\Sigma_{\hat{h}}$, $\Sigma_{\hat{e}}$ are obtained via first-order error propagation, i.e.

$$\Sigma_\chi = \frac{\partial\chi}{\partial\mathrm{vec}\overline{Y}}\,\Sigma_{\mathrm{vec}\overline{Y}}\left(\frac{\partial\chi}{\partial\mathrm{vec}\overline{Y}}\right)' \qquad \chi = \hat{h}, \hat{e}$$

Tracking for elliptical objects

Recall

$$\Sigma_{\hat{h}} = \frac{1}{m-1}\frac{\lambda_1\,\lambda_2}{(\lambda_1-\lambda_2)^2} \qquad \Sigma_{\hat{\ell}} = \frac{1}{m-1}\frac{k}{2}\begin{bmatrix} \frac{\lambda_1^2}{\lambda_1-\sigma_v^2} & \frac{\lambda_1\lambda_2}{2\sqrt{(\lambda_1-\sigma_v^2)(\lambda_2-\sigma_v^2)}} \\ * & \frac{\lambda_2^2}{\lambda_2-\sigma_v^2} \end{bmatrix}$$

Tracking for elliptical objects

Recall

$$\Sigma_{\hat{h}} = \frac{1}{m-1}\frac{\lambda_1\,\lambda_2}{(\lambda_1-\lambda_2)^2} \qquad \Sigma_{\hat{\ell}} = \frac{1}{m-1}\frac{k}{2}\begin{bmatrix} \frac{\lambda_1^2}{\lambda_1-\sigma_v^2} & \frac{\lambda_1\lambda_2}{2\sqrt{(\lambda_1-\sigma_v^2)(\lambda_2-\sigma_v^2)}} \\ * & \frac{\lambda_2^2}{\lambda_2-\sigma_v^2} \end{bmatrix}$$

Implicit assumptions in Extended Object Tracking:

$$\lambda_1 \gg \lambda_2 \gg \sigma_v^2$$

We can consider the margins $\lambda_1 - \lambda_2$, $|\lambda_2 - \sigma_v^2|$ as **quality indicators** of the Signal-to-Noise Ratio (SNR) characterizing the point cloud.

Tracking for elliptical objects

Augmented $\Lambda : O$ state x and static estimate $\mathbb{Y}$

$$x \triangleq \begin{bmatrix} p' & \ell' \end{bmatrix}' \qquad \ell \triangleq \begin{bmatrix} h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} & e' \end{bmatrix}'$$

$$\mathbb{Y} \triangleq \begin{bmatrix} \hat{p}' & \hat{h} & \hat{e}' \end{bmatrix}'$$

Tracking for elliptical objects

Augmented $\Lambda : O$ state x and static estimate $\mathbb{Y}$

$$x \triangleq \begin{bmatrix} p' & \ell' \end{bmatrix}' \qquad \ell \triangleq \begin{bmatrix} h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} & e' \end{bmatrix}'$$

$$\mathbb{Y} \triangleq \begin{bmatrix} \hat{p}' & \hat{h} & \hat{e}' \end{bmatrix}'$$

Prediction equations (random walk for e)

$$x_{k|k-1} \triangleq \overline{f}_{k|k-1}$$

$$P_{k|k-1} \triangleq F_{k|k-1} + Q$$

Tracking for elliptical objects

Augmented $\Lambda : O$ state x and static estimate $\mathbb{Y}$

$$x \triangleq \begin{bmatrix} p' & \ell' \end{bmatrix}' \qquad \ell \triangleq \begin{bmatrix} h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} & e' \end{bmatrix}'$$

$$\mathbb{Y} \triangleq \begin{bmatrix} \hat{p}' & \hat{h} & \hat{e}' \end{bmatrix}'$$

Prediction equations (random walk for e)

$$x_{k|k-1} \triangleq \overline{f}_{k|k-1}$$

$$P_{k|k-1} \triangleq F_{k|k-1} + Q$$

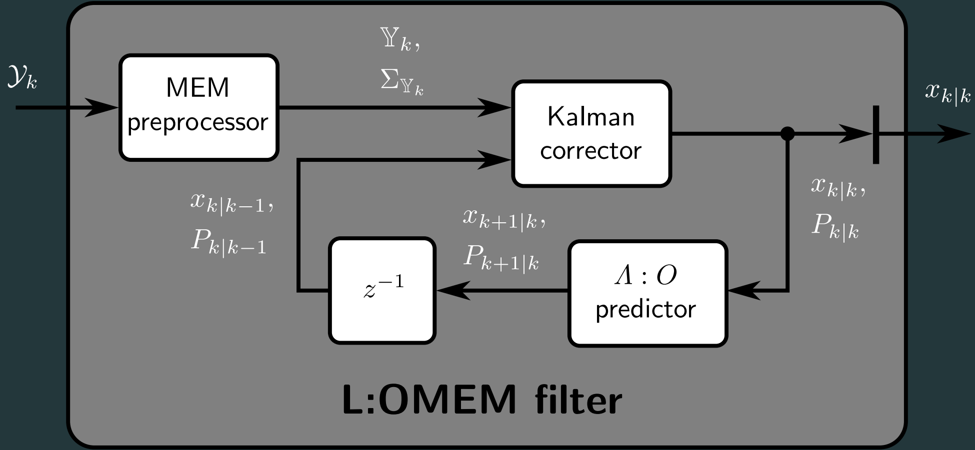
Correction equations

$$L_k = P_{k|k-1}H'\left(HP_{k|k-1}H' + \Sigma_{\mathbb{Y}_k}\right)^{-1}$$

$$x_{k|k} = (I - L_k H)x_{k|k-1} + L_k\,\mathbb{Y}_k$$

$$P_{k|k} = (I - L_k H)P_{k|k-1}$$

Tracking for elliptical objects



Preprocessing has 2 main advantages over conventioal approaches:

- **computational efficiency:** instead of processing m points sequentially ($\mathcal{O}(m)$) or processing a single stack of m points ($\mathcal{O}(m^3)$), preprocessing reduces the correction to $\mathcal{O}(1)$;

Tracking for elliptical objects

Preprocessing has 2 main advantages over conventioal approaches:

- **computational efficiency:** instead of processing m points sequentially ($\mathcal{O}(m)$) or processing a single stack of m points ($\mathcal{O}(m^3)$), preprocessing reduces the correction to $\mathcal{O}(1)$;
- **white box correction:** the static estimate $\mathbb{Y}_k$ is a subset of the object state x_k and not a a nonlinear function $h(x)$ of it (as in RMM and RHM). Hence, we have "maximum correlation" $\Sigma_{x\mathbb{Y}}$ between observation $\mathbb{Y}_k$ and state x_k.

Tracking for general objects

Tracking for general objects

Elliptic models are great for several reasons:

- Easy to implement and, more importantly, computationally cheap;
- Allows for closed-form Bayesian updates
  (+ simple multi-object, multi-sensor extensions);
- They can classify objects with well-distinguished extensions.

Tracking for general objects

Elliptic models are great for several reasons:

- Easy to implement and, more importantly, computationally cheap;
- Allows for closed-form Bayesian updates
  (+ simple multi-object, multi-sensor extensions);
- They can classify objects with well-distinguished extensions.

However, in some scenarios they are deemed to fail:

- When we have to distinguish objects with similar extensions;
- When we have to deal with **occlusions**.

Moreover, ellipses are symmetric: they cannot distinguish bow/front from stern/rear.

Tracking for general objects

TODO: Elliptic fails picture

Tracking for general objects

QUESTION: how do we overcome the limitations of elliptic models?

Tracking for general objects

QUESTION: how do we overcome the limitations of elliptic models?

In many scenarios, we have **strong prior knowledge** about the object being tracked.

Tracking for general objects

QUESTION: how do we overcome the limitations of elliptic models?

In many scenarios, we have **strong prior knowledge** about the object being tracked. We may know it is a car, ship, or aircraft; but not its specific manufacturing model.

Tracking for general objects

QUESTION: how do we overcome the limitations of elliptic models?

In many scenarios, we have **strong prior knowledge** about the object being tracked. We may know it is a car, ship, or aircraft; but not its specific manufacturing model.

IDEA: tackle shape estimation as a **classification** problem
(rather than a regression problem)

Tracking for general objects

QUESTION: how do we overcome the limitations of elliptic models?

In many scenarios, we have **strong prior knowledge** about the object being tracked. We may know it is a car, ship, or aircraft; but not its specific manufacturing model.

IDEA: tackle shape estimation as a **classification** problem (rather than a regression problem)

...moreover, it would be nice to keep the good features of elliptic models.

Tracking for general objects

QUESTION: how do we overcome the limitations of elliptic models?

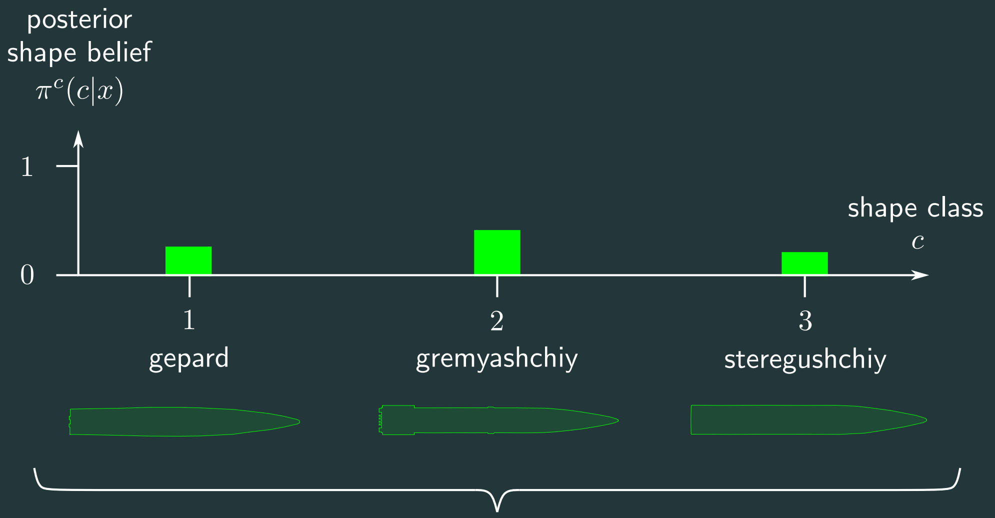
In many scenarios, we have **strong prior knowledge** about the object being tracked. We may know it is a car, ship, or aircraft; but not its specific manufacturing model.

IDEA: tackle shape estimation as a **classification** problem
(rather than a regression problem)

...moreover, it would be nice to keep the good features of elliptic models.

Assumption: we have at disposal a **shape library** of C known "shapes" $c = 1, .., C$.

Tracking for general objects



shape library (e.g. Russian navy from Wikipedia and blueprints)

Tracking for general objects

Why not using Random Hypersurface Models?

Tracking for general objects

Why not using Random Hypersurface Models?

- RHMs handle only star-convex shapes.
- RHM-based filters employ Kalman filters (EKFs, UKFs) including in the state vector n Fourier coefficients, or n radius points, or n vertex positions.

$$\text{RHM regression:} \quad \mathcal{O}(n^3)$$

- Typically, we use the estimated shape by RHM filters to classify tracked objects. Why not perform classification directly over point clouds?

Tracking for general objects

Hybrid L:OMEM state

$$\boldsymbol{x} \triangleq \begin{bmatrix} x' & c \end{bmatrix}' \qquad x \triangleq \begin{bmatrix} p' & h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} & e' \end{bmatrix}'$$

$$c \in \{1, \ldots, C\}$$

Joint tracking and classification belief

$$\pi(\boldsymbol{x}) \triangleq \pi(x, c) = \underbrace{\pi^x(x)}_{\text{kinematic belief}} \; \underbrace{\pi^c(c|x)}_{\text{shape belief}}$$

[1] not necessarily L:OMEM

Tracking for general objects

Hybrid L:OMEM state

$$\boldsymbol{x} \triangleq \begin{bmatrix} x' & c \end{bmatrix}' \qquad x \triangleq \begin{bmatrix} p' & h & s & \cdots & s^{(\Lambda-1)} & \omega & \cdots & \omega^{(O-1)} & e' \end{bmatrix}'$$

$$c \in \{1, \ldots, C\}$$

Joint tracking and classification belief

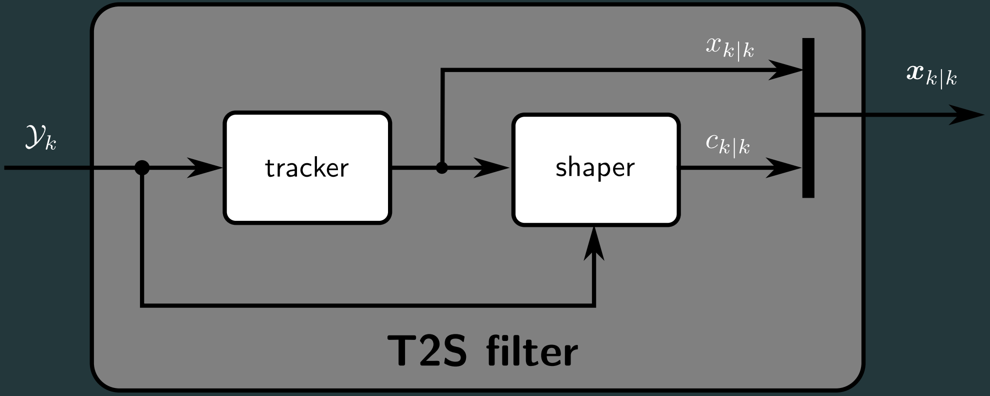
$$\pi(\boldsymbol{x}) \triangleq \pi(x, c) = \underbrace{\pi^x(x)}_{\text{kinematic belief}} \quad \underbrace{\pi^c(c|x)}_{\text{shape belief}}$$

Track-to-Shape (T2S) filter

- employs a *tracker*[1] to update $\pi^x(x)$ according to data;
- employs a *shaper* to update $\pi^c(c|x)$ according to data .

[1] not necessarily L:OMEM

Tracking for general objects



Tracking for general objects

QUESTION: what is the ”shape” of an object?

Tracking for general objects

QUESTION: what is the "shape" of an object?

A reasonable definition of "shape" should uniquely depend on the object geometry.

Tracking for general objects

QUESTION: what is the "shape" of an object?

A reasonable definition of "shape" should uniquely depend on the object geometry.
We look for a definition that is:

- invariant to translation;
- invariant to rotation;
- invariant to scale.

and generalizes the elliptic model and the RHM model.

Tracking for general objects

QUESTION: what is the "shape" of an object?

A reasonable definition of "shape" should uniquely depend on the object geometry.
We look for a definition that is:

- invariant to translation;
- invariant to rotation;
- invariant to scale.

and generalizes the elliptic model and the RHM model.

Accordingly, we define the *object shape* $\widetilde{S}$ as a closed and non self-intersecting polygon (contour or surface) contained in the unit square $[-0.5, +0.5]^2$.
Such polygon is defined by a **shape vector** $\widetilde{S}$ stacking vertex coordinates.

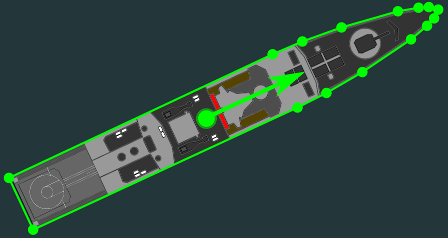
Linear Spline Model

contour equation

$$z(\alpha) \triangleq p + U(h)\, D(e)\, B(\alpha)\widetilde{S}$$

parameters

position	$p \in \mathbb{R}^2$
heading	$h \in [-\pi, \pi)$
semi-length	$a \in \mathbb{R}_{>0}$
semi-width	$b \in \mathbb{R}_{>0}$



Filters

T2S (Track-to-Shape)

TNS (Track-and-Shape)

shape vector $\widetilde{S} \triangleq \left[\ \widetilde{V}_1' \quad \cdots \quad \widetilde{V}_r'\ \right]' \in \mathbb{R}^{2r}$

Tracking for general objects

Since the shape vectors $\{\widetilde{S}^{(c)}\}_{c=1}^{C}$ are referred to the unit square $[-0.5, +0.5]^2$, we need to **whiten** the measurements before feeding them to the shaper.

Tracking for general objects

Since the shape vectors $\{\widetilde{S}^{(c)}\}_{c=1}^{C}$ are referred to the unit square $[-0.5, +0.5]^2$, we need to **whiten** the measurements before feeding them to the shaper.

This is an operation based on the output of the tracker

$$\widetilde{\mathcal{Y}} \triangleq \{\widetilde{y}^{(j)}\}_{j=1}^{m} \qquad \widetilde{y}^{(j)} \triangleq \left(U(\hat{h})\, D(\hat{e})\right)^{-1} \left(y^{(j)} - \hat{p}\right)$$

Tracking for general objects

Since the shape vectors $\{\widetilde{S}^{(c)}\}_{c=1}^{C}$ are referred to the unit square $[-0.5, +0.5]^2$, we need to **whiten** the measurements before feeding them to the shaper.

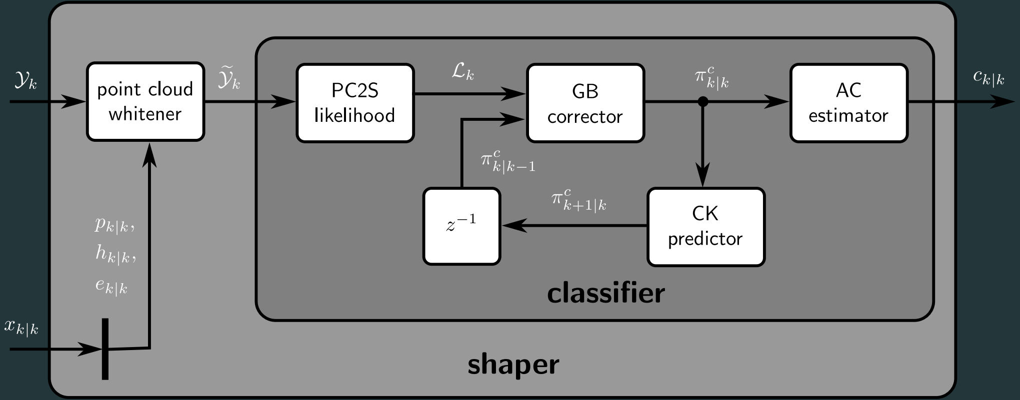
This is an operation based on the output of the tracker

$$\widetilde{\mathcal{Y}} \triangleq \{\widetilde{y}^{(j)}\}_{j=1}^{m} \qquad \widetilde{y}^{(j)} \triangleq \left(U(\hat{h})\,D(\hat{e})\right)^{-1}\left(y^{(j)} - \hat{p}\right)$$

Once whitened, the pointcloud can be compared to the shapes in the library via a **Bayesian classifier**, composed of:

- **(1)** an **Anti-Chattering** (AC) estimator.
- **(2)** a **Chapman-Kolmogorov** (CK) prediction step based on some suitable transition matrix;
- **(3)** a **Generalized Bayesian** (GB) correction step based on some suitable Pointcloud-to-Shape (PC2S) likelihood function;

Tracking for general objects



Notations

$$\pi^c \triangleq \begin{bmatrix} \pi^c(1|x) & \cdots & \pi^c(C|x) \end{bmatrix}'$$

$$\mathcal{L} \triangleq \operatorname{diag}\left(\mathcal{L}(\widetilde{\mathcal{Y}}|\widetilde{S}^{(1)}), \ldots, \mathcal{L}(\widetilde{\mathcal{Y}}|\widetilde{S}^{(C)})\right)$$

Notations

$$\pi^c \triangleq \begin{bmatrix} \pi^c(1|x) & \cdots & \pi^c(C|x) \end{bmatrix}'$$

$$\mathcal{L} \triangleq \operatorname{diag}\left(\mathcal{L}(\widetilde{\mathcal{Y}}|\widetilde{S}^{(1)}), \ldots, \mathcal{L}(\widetilde{\mathcal{Y}}|\widetilde{S}^{(C)})\right)$$

Chapman-Kolmogorov prediction

$$\pi^c_{k|k-1} = \mathcal{T}\,\pi^c_{k-1|k-1}$$

for a suitable transition matrix $\mathcal{T}$.

Notations

$$\pi^c \triangleq \begin{bmatrix} \pi^c(1|x) & \cdots & \pi^c(C|x) \end{bmatrix}'$$

$$\mathcal{L} \triangleq \operatorname{diag}\left(\mathcal{L}(\widetilde{\mathcal{Y}}|\widetilde{S}^{(1)}), \ldots, \mathcal{L}(\widetilde{\mathcal{Y}}|\widetilde{S}^{(C)})\right)$$

Chapman-Kolmogorov prediction

$$\pi^c_{k|k-1} = \mathcal{T}\, \pi^c_{k-1|k-1}$$

for a suitable transition matrix $\mathcal{T}$.

Generalized Bayesian correction

$$\pi^c_{k|k} \propto \exp\left(-J_k\right)\, \pi^c_{k|k-1}$$

$$J_k \triangleq -\frac{1}{\tau_c} \log \mathcal{L}_k$$

for a suitable *temperature* parameter $\tau_c > 0$ and a suitable PC2S likelihood matrix $\mathcal{L}$.

(1) AC estimator

$$c_{k|k} \triangleq \arg\max_{c} \frac{1}{1+\eta} \pi^{c}_{k|k}(c|x) + \frac{\eta}{1+\eta}\, \delta_{c_{k-1|k-1}}(c)$$

where $\eta > 0$ is the hysteresis amplitude.

This estimator smooths out changes in the Maximum A Posteriori (MAP) estimate:

- if $\eta \to 0$, we recover the standard MAP estimator (no smoothing);
- if $\eta \to +\infty$, we recover a zero-order hold estimator (no change).

(2) Transition matrix

$$\mathcal{T} \triangleq (1-\lambda)\,\mathcal{D} + \lambda\,\mathcal{R}$$

where $\lambda \in (0,1)$ is a forgetting factor and:

- **dissimilarity matrix**

  $$[\mathcal{D}]_{ij} \propto \exp\left[-\frac{1}{\tau_\mathrm{p}}\mathrm{dissim}\left(\widetilde{S}^{(i)}, \widetilde{S}^{(j)}\right)\right]$$

  where $\tau_\mathrm{p} > 0$ is a temperature parameter.
  This term makes the classifier robust against geometric ambiguities between similar shapes. Dissimilarity metrics: Hausdorff, chamfer, earth mover, etc...

- **regularization matrix**

  $$[\mathcal{R}]_{ij} \triangleq \frac{1}{C}$$

  This term makes the classifier robust against underflow issues.

(3) PC2S likelihood

Assuming $\widetilde{\mathcal{Y}}$ is as an **Independent and Identically Distributed Cluster (IIDC)** Random Finite Set,

$$\mathcal{L}\left(\widetilde{\mathcal{Y}}\,|\,\widetilde{S}^{(c)}\right) \triangleq \mathcal{L}^{\mathrm{C}}\left(|\widetilde{\mathcal{Y}}|\,|\,\widetilde{S}^{(c)}\right) \prod_{\widetilde{y}\in\widetilde{\mathcal{Y}}} \mathcal{L}^{\mathrm{S}}\left(\widetilde{y}\,|\widetilde{S}^{(c)}\right)$$

where:

- $\mathcal{L}^{\mathrm{C}}\left(|\widetilde{\mathcal{Y}}|\,|\,\widetilde{S}^{(c)}\right)$ is the **cardinality likelihood**.
  It provides a cheap pre-screening of unlikely shapes based on the number of points in the cloud;
- $\mathcal{L}^{\mathrm{S}}\left(\widetilde{y}\,|\widetilde{S}^{(c)}\right)$ is the **spatial likelihood**.
  It provides a deep analysis of the compatibility between each point in the cloud and the shape under test.

Tracking for general objects

To define, $\mathcal{L}^{\mathrm{C}}\left(|\widetilde{\mathcal{Y}}| \,\middle|\, \widetilde{S}^{(c)}\right)$ and $\mathcal{L}^{\mathrm{S}}\left(\widetilde{y} \,\middle|\, \widetilde{S}^{(c)}\right)$, we need to introduce:

- **shape patch**: the object shape is decomposed into n non-overlapping patches

$$\widetilde{\mathcal{S}} = \bigcup_{i=1}^{n} \widetilde{\mathcal{S}}_i$$

If $\widetilde{\mathcal{S}}$ is the polygon contour, $\widetilde{\mathcal{S}}_i$ is the i-th polygon edge.
If $\widetilde{\mathcal{S}}$ is the polygon surface, $\widetilde{\mathcal{S}}_i$ is the i-th polygon triangle.

Tracking for general objects

To define, $\mathcal{L}^{\mathrm{C}}\left(|\widetilde{\mathcal{Y}}| \,\middle|\, \widetilde{S}^{(c)}\right)$ and $\mathcal{L}^{\mathrm{S}}\left(\widetilde{y} \,\middle|\, \widetilde{S}^{(c)}\right)$, we need to introduce:

- **shape patch**: the object shape is decomposed into n non-overlapping patches

$$\widetilde{\mathcal{S}} = \bigcup_{i=1}^{n} \widetilde{\mathcal{S}}_i$$

  If $\widetilde{\mathcal{S}}$ is the polygon contour, $\widetilde{\mathcal{S}}_i$ is the i-th polygon edge.
  If $\widetilde{\mathcal{S}}$ is the polygon surface, $\widetilde{\mathcal{S}}_i$ is the i-th polygon triangle.

- **patch measure**: to each patch $\widetilde{\mathcal{S}}_i$ we associate a measure

$$\mu_i \triangleq \mu\left(\widetilde{\mathcal{S}}_i\right)$$

  If $\widetilde{\mathcal{S}}$ is the polygon contour, μ_i is the **length** of the i-th polygon edge.
  If $\widetilde{\mathcal{S}}$ is the polygon surface, μ_i is the **area** of the i-th polygon triangle.

Tracking for general objects

TODO picture with shape patches and patch measures

Cardinality likelihood

$$\mathcal{L}^{\mathrm{C}}\left(|\widetilde{\mathcal{Y}}| \mid \widetilde{S}^{(c)}\right) \triangleq \mathrm{Poisson}\left(|\widetilde{\mathcal{Y}}|;\, \lambda^{(c)}\right)$$

$$\lambda^{(c)} \triangleq \rho\, a^{(c)}\, b^{(c)} \sum_{i=1}^{n^{(c)}} \mu_i^{(c)}$$

where:

- $\rho > 0$ is the **sensor resolution** (expected number of points per unit measure);
- $a^{(c)}, b^{(c)}$ are the object length and width (from shape library).

IDEA: "bigger" is the shape, larger is the number of points we expect in the cloud.

Spatial likelihood

assuming that the point cloud is uniformly distributed over the object shape $\mathcal{S}$,

$$\mathcal{L}^{\mathrm{S}}\left(\widetilde{y} \,\middle|\, \widetilde{S}^{(c)}\right) \triangleq \sum_{i=1}^{n^{(c)}} \frac{\mu_i^{(c)}}{\mu^{(c)}} \int_{[0,1]^d} \pi_v\left(\widetilde{y} - \widetilde{\kappa}_i^{(c)}(\beta)\right) \, \mathsf{d}\beta$$

$$\mu^{(c)} \triangleq \sum_{i=1}^{n^{(c)}} \mu_i^{(c)}$$

where $d = 1$ in the contour case, $d = 2$ in the surface case, and:

- $\pi_v(\cdot)$ is the sensor noise density;
- $\widetilde{\kappa}_i^{(c)}(\cdot)$ is the **patch covering kernel**.

Spatial likelihood

assuming that the point cloud is uniformly distributed over the object shape $\mathcal{S}$,

$$\mathcal{L}^{\mathrm{S}}\left(\widetilde{y}\,\middle|\,\widetilde{S}^{(c)}\right) \triangleq \sum_{i=1}^{n^{(c)}} \frac{\mu_i^{(c)}}{\mu^{(c)}} \int_{[0,1]^d} \pi_v\left(\widetilde{y} - \widetilde{\kappa}_i^{(c)}(\beta)\right)\, \mathsf{d}\beta$$

$$\mu^{(c)} \triangleq \sum_{i=1}^{n^{(c)}} \mu_i^{(c)}$$

where $d = 1$ in the contour case, $d = 2$ in the surface case, and:

- $\pi_v(\cdot)$ is the sensor noise density;
- $\widetilde{\kappa}_i^{(c)}(\cdot)$ is the **patch covering kernel**.

If $p_v(\cdot)$ is Gaussian, the integral can be computed in closed form in the contour case.
In the surface case, the integral can be computed via numerical techniques (e.g. MC).

Tracking for general objects

TODO kernel pictures

Tracking for general objects

T2S is **linear** in the shape complexity $n^{(c)}$:

$$\mathcal{L}\left(\widetilde{\mathcal{Y}} \,\middle|\, \widetilde{S}^{(c)}\right) = \underbrace{\underbrace{\text{Poisson}\left(|\widetilde{\mathcal{Y}}|;\, \lambda^{(c)}\right)}_{\mathcal{O}(1)} \prod_{j=1}^{m} \sum_{i=1}^{n^{(c)}} \frac{\mu_i^{(c)}}{\mu^{(c)}} \underbrace{\int_{[0,1]^d} \pi_v\left(\widetilde{y}^{(j)} - \widetilde{\kappa}_i^{(c)}(\beta)\right) \, \mathsf{d}\beta}_{\mathcal{O}(N_i^{(c)})}}_{\mathcal{O}(mn^{(c)}\bar{N}^{(c)})}$$

Tracking for general objects

T2S is **linear** in the shape complexity $n^{(c)}$:

$$\mathcal{L}\left(\widetilde{\mathcal{Y}} \mid \widetilde{S}^{(c)}\right) = \underbrace{\underbrace{\text{Poisson}\left(|\widetilde{\mathcal{Y}}|;\, \lambda^{(c)}\right)}_{\mathcal{O}(1)} \prod_{j=1}^{m} \sum_{i=1}^{n^{(c)}} \frac{\mu_i^{(c)}}{\mu^{(c)}} \underbrace{\int_{[0,1]^d} \pi_v\left(\widetilde{y}^{(j)} - \widetilde{\kappa}_i^{(c)}(\beta)\right)\, \mathrm{d}\beta}_{\mathcal{O}(N_i^{(c)})}}_{\mathcal{O}(mn^{(c)}\bar{N}^{(c)})}$$

The overall complexity is $\mathcal{O}(Cm\bar{n}\bar{N})$:

- C is the number of shapes in the shape library;
- m is the number of points in the cloud;
- $\bar{n}$ is the average number of patches across the shape library;
- $\bar{N}$ is the average number of MC particles across the shape library.

Conclusions

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;
- shape is time-invariant for rigid objects;

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;
- shape is time-invariant for rigid objects;
- shape does not necessarily have to be generated ex-novo.

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;
- shape is time-invariant for rigid objects;
- shape does not necessarily have to be generated ex-novo.

The **main limitations** of the proposed solution are:

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;
- shape is time-invariant for rigid objects;
- shape does not necessarily have to be generated ex-novo.

The **main limitations** of the proposed solution are:

- accuracy significantly drops with non-uniform point clouds;

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;
- shape is time-invariant for rigid objects;
- shape does not necessarily have to be generated ex-novo.

The **main limitations** of the proposed solution are:

- accuracy significantly drops with non-uniform point clouds;
- high computational cost;

Summary

A solution for EOT has been presented, **taking into account the following aspects**:

- point cloud provides information about heading;
- shape is time-invariant for rigid objects;
- shape does not necessarily have to be generated ex-novo.

The **main limitations** of the proposed solution are:

- accuracy significantly drops with non-uniform point clouds;
- high computational cost;
- shape does not affect position and heading estimation.

Outlook

- **Direction 1:** occlusion-based EOT via ray-casting

Outlook

- **Direction 1:** occlusion-based EOT via ray-casting
- **Direction 2:** multisensor EOT (centralized and distributed)

Outlook

- **Direction 1:** occlusion-based EOT via ray-casting
- **Direction 2:** multisensor EOT (centralized and distributed)
- **Direction 3:** multi-EOT in clutter via Random Finite Sets

Outlook

- **Direction 1:** occlusion-based EOT via ray-casting
- **Direction 2:** multisensor EOT (centralized and distributed)
- **Direction 3:** multi-EOT in clutter via Random Finite Sets
- **Direction 4:** agnostic shaping via MAP optimization and deep learning

Outlook

- **Direction 1:** occlusion-based EOT via ray-casting
- **Direction 2:** multisensor EOT (centralized and distributed)
- **Direction 3:** multi-EOT in clutter via Random Finite Sets
- **Direction 4:** agnostic shaping via MAP optimization and deep learning
- **Direction 5:** 3-dimensional EOT via computer vision models

Thank you

"The first principle is that you must not fool yourself,
and you are the easiest person to fool."
— *Richard Feynman*