

# The *CyCab*: Bayesian navigation on sensory–motor trajectories

Cédric Pradalier<sup>1</sup>

Pierre Bessière<sup>2</sup>

March 25, 2008

## 1 Introduction

Autonomous navigation of a mobile robot is a widely studied problem in the robotics community. Most robots designed for this task are equipped with onboard sensor(s) to perceive the external world (sonars, laser telemeters, camera). Two main approaches to autonomous navigation have been proposed: reactive navigation, where the robot uses only its current perceptions to move and explore without colliding (e.g. Arkin [1998] or Bonasso et al. [1995]), and servoed navigation, in which the robot is given a preplanned reference trajectory and uses some closed-loop control law to follow it (e.g. Laumond et al. [1989] or Lamiraux et al. [1999]). In servoed problems, two classes of approaches can again be separated: state-space tracking (e.g. Hermosillo et al. [2003b,a]) and perception- space tracking (e.g. Malis et al. [2001] or Chaumette [1994]).

State-space tracking implies two capabilities: first, to be given a reference trajectory in the state space, and second, to be able to localize the robot, also in the state space. Conversely, perception-space tracking implies that the trajectory is defined with respect to perception only, hence avoiding the need for global localization. A specific application of perception tracking is visual servoing, classically implemented as the convergence of the observed image to a fixed reference image.

In this chapter, we are specifically interested in the case of perceptual tracking of a sensory–motor trajectory (**SMT**) with a mobile robot. We assume that:

- the reference trajectory is defined as a sequence of observations perceived by onboard sensors while the robot is moving; and
- no localization system (either GPS or landmark-based) is available to perform tracking.

This situation is interesting for at least three reasons: first, because the trajectory is not defined with respect to a Cartesian frame, we can ignore the complex task of global localization; second, such trajectories can be naturally and easily learnt from examples; and third, this approach can be seen as a hypothesis on how biological entities memorize and represent paths.

This chapter is organized as follows. Sections 2 and 3 introduce the notions and definitions used in this article. The elementary modules required for safe navigation on a sensory–motor trajectory are developed in Sections 4 to 7. Finally, Section 8 presents some experiments on a simulated platform and some results on our real platform: the *CyCab*.

## 2 Problem statement

### 2.1 General scenario

The application we consider in this paper has two stages.

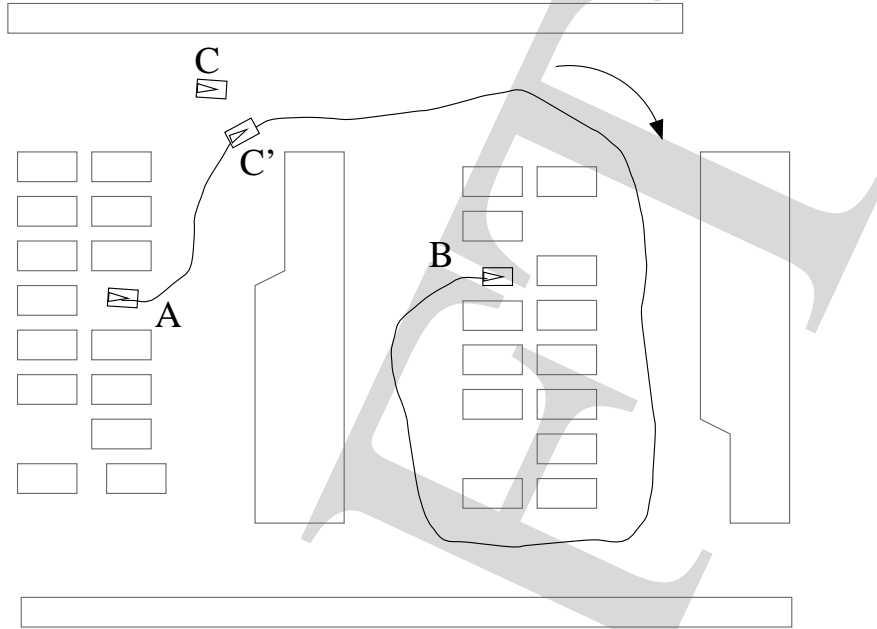
- Learning an SMT: the robot is driven manually from a configuration  $A$  to a configuration  $B$ . While it is driven, it records its perceptions from onboard sensors (e.g. a camera or a laser range finder) and the commands sent by the driver.
- Navigating on the SMT: we now assume that the robot starts in a configuration  $C$  in the neighbourhood of the SMT. Our goal is then to reach configuration  $B$  while following the SMT as closely as possible.

Figure 1 gives an intuitive representation of our objectives for a simulated vehicle in a car park.

### 2.2 Definitions

**Definition 1. Sensory–motor trajectory (SMT):** *Given two sets  $\mathbf{O}$  and  $\mathbf{U}$  containing respectively the perceptions and the actions of the robot, we define a sensory–motor trajectory  $\mathcal{T}_{sm}$  as a function of time with values in the Cartesian product of the robot command space  $\mathbf{U}$  and its observation space  $\mathbf{O}$ . Formally:*

$$\begin{aligned} \mathcal{T}_{sm} : [0, t_{\max}] &\longrightarrow \mathbf{O} \times \mathbf{U} \\ t &\mapsto (\mathcal{T}_{sm}(t).Z, \mathcal{T}_{sm}(t).U). \end{aligned}$$



**Fig. 1.** Navigating on the SMT from  $A$  to  $B$ : starting from  $C$ , the objective is to follow the black track to reach  $B$ .

**Definition 2. Temporal position:** Consider an SMT  $\mathcal{T}_{sm}$ . The temporal position  $\tau^t$  is a time in  $[0, t_{\max}]$  such that, at time  $t$ ,  $\mathcal{T}_{sm}(\tau^t).Z$  and  $\mathcal{T}_{sm}(\tau^t).U$  are considered respectively as reference observation and reference action by a robot following the SMT.

**Definition 3. Difference of viewpoint:** Let  $Z_1$  be an observation received in configuration  $C_1$ , and let  $Z_2$  be an observation received in  $C_2$ . We define the difference of viewpoint between  $Z_1$  and  $Z_2$  as the difference of configuration  $C_2 - C_1$ .

**Definition 4. Tracking error:** Consider an SMT  $\mathcal{T}_{sm}$  and an observation  $Z^t$ . The tracking error  $\xi^t$  is the difference of viewpoint between  $Z^t$  and the reference observation  $\mathcal{T}_{sm}(\tau^t).Z$ .

### 2.3 Problems to solve

To navigate on a sensory–motor trajectory, our robot must be equipped with several core competencies:

1. it must be able to localize itself on the trajectory;
2. it must be able to control its movement to follow the trajectory while avoiding collisions with unexpected obstacles; and
3. it must be able to estimate online its confidence with respect to its localization hypotheses.

### 3 Outline of the approach

To solve these problems, we implement a set of specialized functions.

- Initialization: when the system starts, we only assume that the robot is in a neighbourhood of the SMT. Consequently, using the first observation  $Z^0$ , we must estimate the initial temporal location  $\tau^0$  and the initial tracking error  $\xi^0$ .
- Localization: during its movement, the system must keep track of the temporal location  $\tau^t$  – *i.e.* temporal localization – and of the tracking error  $\xi^t$  – *i.e.* egocentric localization. To this end, the system compares executed commands and real observations with their reference counterparts.
- Command generation: computation of the vehicle controls for tracking the SMT.

In this chapter, we will show how to implement all these basic competencies using Bayesian modelling. Bayesian reasoning often raises some fears when dealing with vehicular robots or heavy machinery: can we guarantee safety and performance when using a method based on probabilities? To answer this question, we want to stress the fact that using Bayesian modelling does not impose

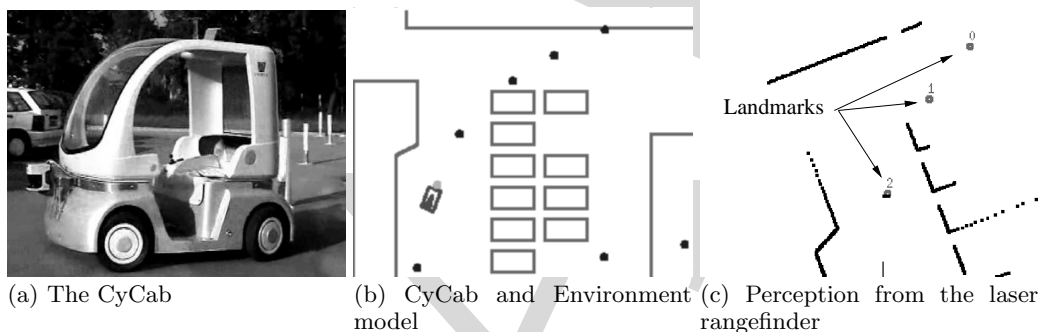
Name	Output	Input
Initialization	$\tau^0, \xi^0$	$Z^0, \mathcal{T}_{sm}$
Localization		
Temporal	$\tau^t$	$Z^t, U^{t-dt}, \mathcal{T}_{sm}$
Relative	$\xi^t$	$Z^t, \mathcal{T}_{sm}(\tau^t).Z$
Command	$U^t$	$\xi^t, \mathcal{T}_{sm}(\tau^t).U$
Confidence	$\text{Conf}^t$	$\xi^t, Z^t, \mathcal{T}_{sm}(\tau^t).Z, \text{Conf}^{t-dt}$

**Table 1.** Specification of the required functions for navigation on a SMT.

non-deterministic behaviours. Bayesian modelling is only a way to express knowledge about the robot, its sensors and its environment. From this knowledge, we can choose to take deterministic or probabilistic decisions.

## 4 Relative localization

### 4.1 Context and vehicle model



**Fig. 2.** CyCab robot and models used throughout this article.

Our general context is inspired by our experimental platform.

#### Vehicle

Our experimental platform is an autonomous electric vehicle (see Fig. 2.a) derived from a golf cab: the CyCab. This 300kg vehicle is capable of speeds up to 4m/s. The model we use represents the CyCab as a rectangular bounding box, in a darker stroke with a triangle inside in Fig. 2.b.

#### Environment

We conducted our experiments in the car park of our institute, among parked or manoeuvring cars and walking pedestrians. Moreover, we assumed that, in this environment, it is sometimes possible to perceive some landmarks. In practice, these landmarks are vertical cylinders covered with reflective sheets. We model this environment by a set of polygonal obstacles (straight lines on Fig. 2.b) and a set of punctual landmarks (scattered discs).

#### Sensors

The CyCab is equipped with a laser rangefinder, mounted on the vehicle's front (see Fig. 2.a and b). Its laser beam sweeps the half-plane in front of the CyCab about 50 centimetres above the ground and returns distances to nearest obstacles and the associated intensity of the returning beam. Figure 2.c illustrates the sensor output in the specific situation depicted in Fig. 2.b. Furthermore, we implemented a landmark extractor that reliably processes the raw data to extract landmark positions in the sensor frame. In the sequel, we will ignore the raw data and consider that **a perception is a set of landmark positions in the sensor frame**. Such a perception is depicted in Fig. 2.c.

This context should not be seen as restricting the application field of our results but rather as an example extended throughout this chapter.

## 4.2 Design of a localization model

A Bayesian model for localization links robot localization and robot perception. Its key part is a sensor model, i.e. a model that can predict expected measurements knowing the robot’s localization. The first part of this section will describe such a model. We will then see how to use it actually to perform the localization.

When a map of the environment is available, it is defined in some fixed frame. The goal of localization is then to compute the robot’s position in this frame.

When a robot is following an SMT, we do not have such a map, but localization with respect to a fixed frame is not required. Because the robot is tracking a trajectory, it simply evaluates its tracking error  $\xi^t$ . It does this by comparing the current observation  $Z^t$  and the reference observation  $\mathcal{T}_{sm}(\tau^t).Z$ .

To simplify notation, we will use  $Z_{ref}^t = \mathcal{T}_{sm}(\tau^t).Z$ . Furthermore, except when explicitly specified otherwise, we will consider all variables to refer to the current time  $t$ . Consequently, we will omit  $t$  whenever possible.

## 4.3 Bayesian sensor model

The sensor model that we now consider is a probability distribution  $P(Z | \xi \wedge Z_{ref})$ . This distribution expresses our knowledge about the expected observation, knowing that the robot is observing  $Z_{ref}$  with a difference of viewpoint  $\xi$ . If this expectation does not contain any uncertainty, then the probability distribution will be a Dirac distribution<sup>1</sup>. Otherwise when we want to express greater uncertainty, the distribution will be flatter.

In the following, we will show the step-by-step construction of the sensor model that we use in our application. Recall the variables’ meanings.

- $\xi$ : the difference of viewpoint between the current observation and the reference one, composed of the  $(x, y)$  position in the plane and the heading  $\theta$ .
- $Z$ , resp.  $Z_{ref}$ : an observation, resp. reference observation, as a set of observed landmark positions in the sensor frame: they can be described either with their polar coordinates  $(\rho, \alpha)$ , or with their Cartesian coordinates  $(x, y)$ .

### Basic model

The simplest model that we can imagine occurs when only one landmark  $L_1$  has been observed in  $Z_{ref}$ . In this case, the sensor model is usually defined as a Gaussian model ( $\mathcal{G}$ ) centred around the distances and bearings that we can expect knowing the difference of viewpoint.

$$P_1([Z = (\rho, \alpha)] | \xi \wedge L_1) = \mathcal{G}(\|(x, y) - L_1\|, \sigma_\rho) \times \mathcal{G}(\arg((x, y) - L_1) - \theta, \sigma_\alpha)$$

This model is parameterized by the standard deviations of the Gaussian distributions  $\sigma_\rho$  and  $\sigma_\alpha$ . Basically, they express the accuracy that we expect from our sensor: larger standard deviations indicate a less accurate sensor.

Using this model, we can now use the Bayesian program in Fig. 3 to answer the question  $P(\xi | Z \wedge Z_{ref})$ .

Figure 4 shows a reference observation, a real observation and the resulting distribution  $P(\xi | Z \wedge Z_{ref})$  for values of  $\xi$  in  $[-5, 5] \times [-5, 5]$  (in meters). To represent this distribution fully, we would require a 4D graph. In the left-hand part of the figure, each arrow represents a value of  $\xi = (x_\xi, y_\xi, \theta_\xi)$ : it starts from  $(x_\xi, y_\xi)$  and has the orientation  $\theta_\xi$ . The length of each arrow is proportional to the likelihood of the tracking error that it represents.

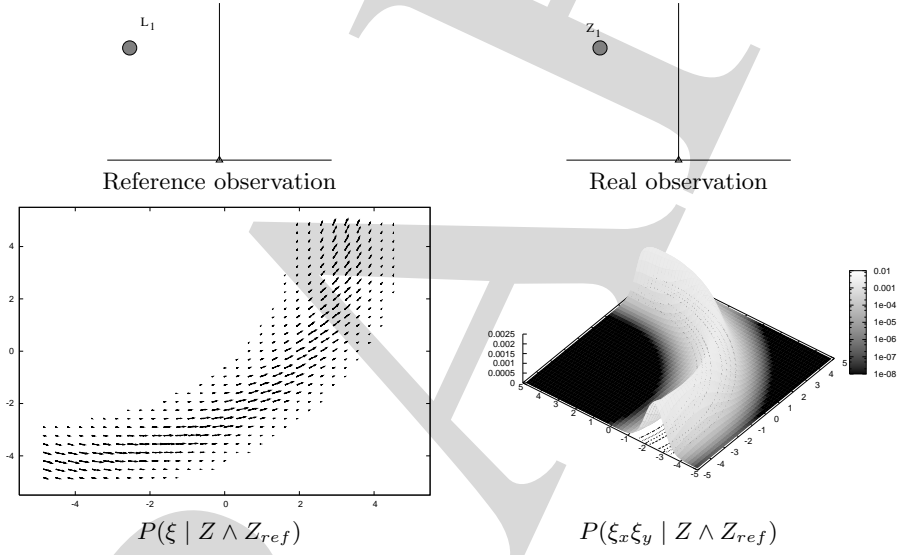
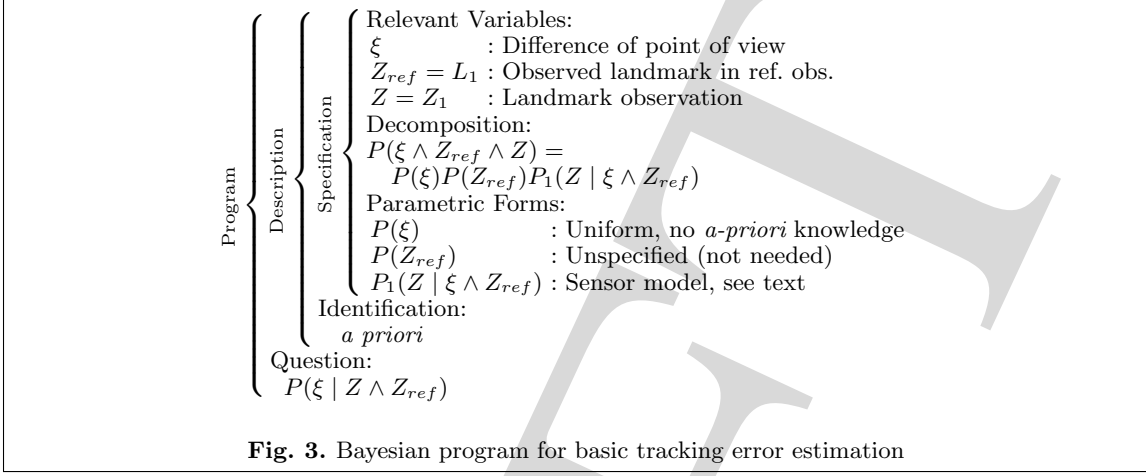
To help visualize the shape of the distributions, we also add the marginalized distribution  $P(\xi_x \wedge \xi_y | Z \wedge Z_{ref})$  on the right-hand part of the figure.

Given one reference landmark and one observation, it is not possible to identify one single difference of point of view. As partly shown in Fig. 4, the set of possible location forms a fuzzy spiral in the 3D configuration space  $(x, y, \theta)$ .

### Taking into account unmodelled events

In our car park environment, with our sensor, it can happen that a vehicle’s light or number plate generates a sensor output similar to a landmark. Such sensor outputs, inconsistent with the sensor model, are often called “outliers” or “false positives”. In the context of probabilistic localization, it is essential to take these into account. Actually, our sensor model gives us both positions that are likely with regard to a given observation and those that are unlikely because of their inconsistency

<sup>1</sup> A Dirac distribution is zero everywhere except for one value.



**Fig. 4.** Building a probabilistic localization: basic model.

with the observation. Consequently, if not dealt with properly, an outlier will classify a whole part of the configuration space as extremely unlikely, including the correct position.

To express the fact that an observation might indeed be an outlier, we add a Boolean variable  $F$  (as **F**alse). When  $F = 1$ , the current observation is assumed to be a false positive. In this case, we cannot expect any peculiar observation. This can be expressed using a uniform distribution. The probabilistic sensor model becomes:

$$P_2([Z = (\rho, \alpha)] | \xi \wedge F \wedge Z_{ref}) = \begin{cases} [F = 0] \rightarrow P_1(Z | \xi \wedge Z_{ref}) \\ [F = 1] \rightarrow \text{Uniform}(\rho) \text{Uniform}(\alpha). \end{cases}$$

In the Bayesian program, the joint distribution becomes:

$$P(Z \wedge \xi \wedge F \wedge Z_{ref}) = P(Z_{ref})P(\xi)P(F)P_2(Z | \xi \wedge F \wedge Z_{ref}). \quad (1)$$

The experimental calibration of  $P(F)$  is a difficult problem because we introduced this term to account for unmodelled terms. Empirically, we chose  $P([F = 1]) = 0.2$ , i.e. we consider that 20% of the observations can be outliers.

### Integrating landmark identification

We now consider the case where several landmarks,  $Z_{ref} = \{L_1, \dots, L_n\}$ , were observed in the reference observation but only one landmark is found in the current observation. The problem is

then to find which of the reference landmarks has been observed. To this end, we introduce a new probabilistic variable  $M \in [1, n]$  (as *Matching*). When  $M = k$ , we expect  $Z$  to be an observation of  $L_k$ . For non-outlier observations, the probabilistic sensor model becomes:

$$P_3([Z = (\rho, \alpha)] \mid \xi \wedge [M = k] \wedge F \wedge Z_{ref}) = P_2(Z \mid \xi \wedge F \wedge L_k). \quad (2)$$

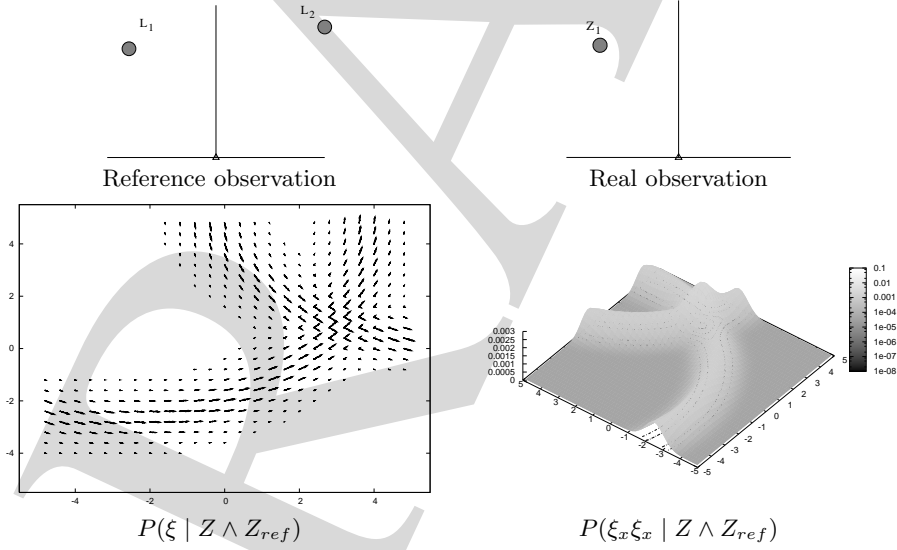
In the Bayesian program, the joint distribution becomes:

$$P(Z \wedge \xi \wedge M \wedge F \wedge Z_{ref}) = P(\xi)P(F)P(M \mid \xi \wedge Z_{ref} \wedge F) \times P(Z_{ref})P_3(Z \mid \xi \wedge M \wedge Z_{ref} \wedge F).$$

When the observation is not an outlier ( $F = 0$ ), the term  $P(M \mid \xi \wedge Z_{ref} \wedge F)$  provides a means of expressing some knowledge about sensor limitations such as range or field of view. Knowing  $\xi$  and  $Z_{ref}$ , a landmark  $L_k$  may be out of range. In this case, we can say that  $P([M = k] \mid \xi \wedge Z_{ref})$  should be small. Nevertheless, in general, we do not want to privilege some identifications and, consequently, we use a uniform distribution over  $M$ .

When the observation is an outlier ( $F = 1$ ), the matching variable is no longer relevant, and  $P(M \mid \xi \wedge Z_{ref} \wedge F)$  can be unspecified. This is made explicit in the complete expression of the localization question:

$$P(\xi \mid Z \wedge Z_{ref}) \propto P(\xi) \times \left[ (P(F = 0) \sum_{k=1}^n P(M = k) P(Z \mid \xi \wedge [F = 0] \wedge [M = k] \wedge Z_{ref})) + P(F = 1) P(Z \mid \xi \wedge [F = 1] \wedge Z_{ref}) \right]$$



**Fig. 5.** Building a probabilistic localization: a model integrating data association and false positives.

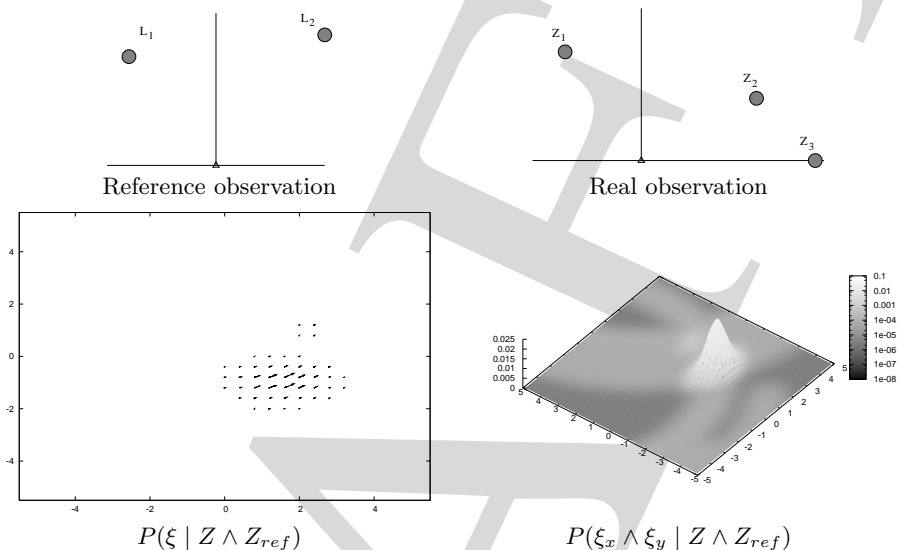
Figure 5 shows a reference observation with two landmarks, a real observation with one landmark, and the resulting distribution  $P(\xi \mid Z \wedge Z_{ref})$  as in Fig. 4. Without knowing the data association, there is no single localization hypothesis but rather two equiprobable spirals in the configuration space, corresponding to the two possible data associations. On the  $P(\xi_x \wedge \xi_y \mid Z \wedge Z_{ref})$  graph, these two spirals are visible as two arcs of circles. We can notice an increase in the likelihood at their intersection. This is because for this position, observation  $Z$  is equally supported by two distinct orientations.

The influence of unmodelled events is also visible on these figures. A configuration's likelihood does not decrease to zero when considering configurations far from the main arcs of circles in the  $P(\xi_x \wedge \xi_y \mid Z \wedge Z_{ref})$  graph: it is bounded below by the configuration likelihood that would result from the observation being a false positive.

## Multiple landmark observations

When reference and real observations contain several landmarks, we must use a set of variables  $Z_i$ ,  $M_i$  and  $F_i$  for each observed landmark. Then, each observation is described with the sensor model presented above. The resulting joint distribution is:

$$P(\xi \mid Z_{1:p} \wedge M_{1:p} \wedge F_{1:p} \wedge Z_{ref}) = P(Z_{ref})P(\xi) \times \prod_{i=1}^p P(F_i)P(M_i \mid \xi \wedge Z_{ref} \wedge F_i)P(Z_i \mid \xi \wedge M_i \wedge Z_{ref} \wedge F_i).$$



**Fig. 6.** Building a probabilistic localization: multiple landmarks model.

Figure 6 shows a two-landmarks reference observation, a three-landmarks real observation and the resulting distribution  $P(\xi \mid Z \wedge Z_{ref})$  as in Fig. 4. After fusing the data from all three observations, a set of robot configurations around  $(1.5, -1, 20^\circ)$  is much more likely than any others. Nevertheless, the configuration spirals that would result from one of the observations being a false positive are still visible, in lighter gray on the  $P(\xi_x \wedge \xi_y \mid Z \wedge Z_{ref})$  graph. If all the observations are false positives, then all tracking errors are equiprobable. The localization hypotheses corresponding to this situation appear in a darker shade on the graph.

It is also interesting to notice that as a false positive, observation  $Z_3$  has little influence on the final distribution. It only introduces a secondary hypothesis, visible as a small set of arrows and a slightly lighter patch behind the main peak on the distribution graph. This hypothesis corresponds to the most likely localization, should  $Z_2$  be an outlier.

### Interest of Bayesian localization

The main interest of the model above is its ability to capture all the localization hypotheses in one computation. After this computation, all the uncertainties on the sensor measure, on the data association and on the presence of outliers are converted into the localization space. In this space, the uncertainty is much easier to handle, especially in the context of some Bayesian filtering. In our application, for instance, the localization is performed by integrating this sensor model in a particle filter.

### Weakness of alternative hypotheses

Using Fig. 6, we have stressed the Bayesian model's ability to extract all the localization hypotheses in one computation. Nevertheless, we should notice that the likelihood of alternative hypotheses is extremely low: several orders of magnitude below the main hypothesis. There are several reasons to try to avoid this situation:

- this low likelihood is not because of the low probability of false-positives but because of the size of the observation space (see below);

- within the context of Bayesian filtering, configurations with such a low likelihood will certainly be discarded; and
- finally, we can wonder if it is really relevant to waste so much computation power to deal with such a small part of the likelihood space.

If we develop the localization equation, we have the following.

$$P(\xi \mid Z \wedge Z_{ref}) \propto \prod_{i=1}^p \left| \begin{array}{l} \underbrace{P([F_i = 0])}_{0.8} \underbrace{P(Z_i \mid \xi \wedge Z_{ref} \wedge [F_i = 0])}_{\text{Gaussian}} \\ + \underbrace{P([F_i = 1])}_{0.2} \underbrace{P(Z_i \mid \xi \wedge Z_{ref} \wedge [F_i = 1])}_{\text{Uniform}} \end{array} \right|$$

Alternative localization hypotheses occur when one of the observations, assume the  $k$ th, is inconsistent with a given position. The Gaussian part of the  $k$ th sum is then close to zero, and only the uniform part remains. Unfortunately, if the observation space is large, the uniform value on this space is very small, and consequently the corresponding position likelihood becomes very small as well, with only a small influence of the false positive distribution.

#### 4.4 Diagnosis model

To build a sensor model that gives a fairer influence to the false-positive variable, we chose to describe our sensor model with a diagnosis variable. Instead of expressing the expected observation knowing the position, we want to use a diagnosis variable to evaluate the consistency between an observation and a position. In practice, this consistency is represented by a Boolean variable  $\mathbb{I}$ .

In the single-landmark/single-observation model with false positives, the consistency model is as follows.

$$P(\mathbb{I} \mid [Z_{ref} = L_1] \wedge Z = (\rho, \alpha) \wedge \xi \wedge F) = \begin{cases} [F = 0] \rightarrow \exp\left(-\frac{1}{2} \frac{(\rho - \|\xi_x, \xi_x - L_1\|)^2}{\sigma_\rho^2}\right) \times \\ \exp\left(-\frac{1}{2} \frac{(\alpha - \arg(\xi_x, \xi_y) - L_1)^2}{\sigma_\alpha^2}\right) \\ [F = 1] \rightarrow 0.5 \end{cases} \quad (3)$$

In this equation,  $\sigma_\rho$  and  $\sigma_\alpha$  act as standard-deviation parameters in Gaussian distributions, expressing the expected accuracy of the sensor. In the true positive case, with  $F = 0$ , as an observation approaches the expected observation, the consistency likelihood will approach 1.

The false-positive case is the interesting part of this model. As in the previous model, we cannot expect any particular observation in this case, and consequently, we choose a consistency likelihood with as much *a-priori* knowledge as possible: 0.5. It may be argued that 0.5 is a high likelihood for the consistency of any conjunction of observation and position. The key advantage of this value is that it does not depend on the size of the observation space.

Figure 7 shows the localization results with a diagnosis model, in the same situation as that shown in Fig. 6. The global shape of the resulting distribution is similar to that in Fig. 6. However, we can observe that alternative localization hypotheses have a much stronger influence: here we do not require a logarithmic colour scale to be able to see the alternative hypotheses. In the context of a Bayesian filtering scheme, this sensor model will consequently be more robust to false positives.

#### 4.5 Remarks on Bayesian localization

We have shown how to build a Bayesian sensor model for localization. Such a model can be used to project all the information available in the observations into the localization space:

- all the data association hypotheses;
- the localization hypotheses accounting for possible false positives;
- the geometrical consistency of the observations;
- the uncertainty on all the above points is now converted into an uncertainty in the localization space, which is much easier to handle, especially in a filtering context.

### 5 System initialization

The first application of our sensor model is the initial localization of the robot in the SMT. Let us recall our objectives: we assume that we know an SMT going from  $A$  to  $B$  and, at system initialization, the robot is at an unknown position  $C$  in the neighbourhood of the trajectory.

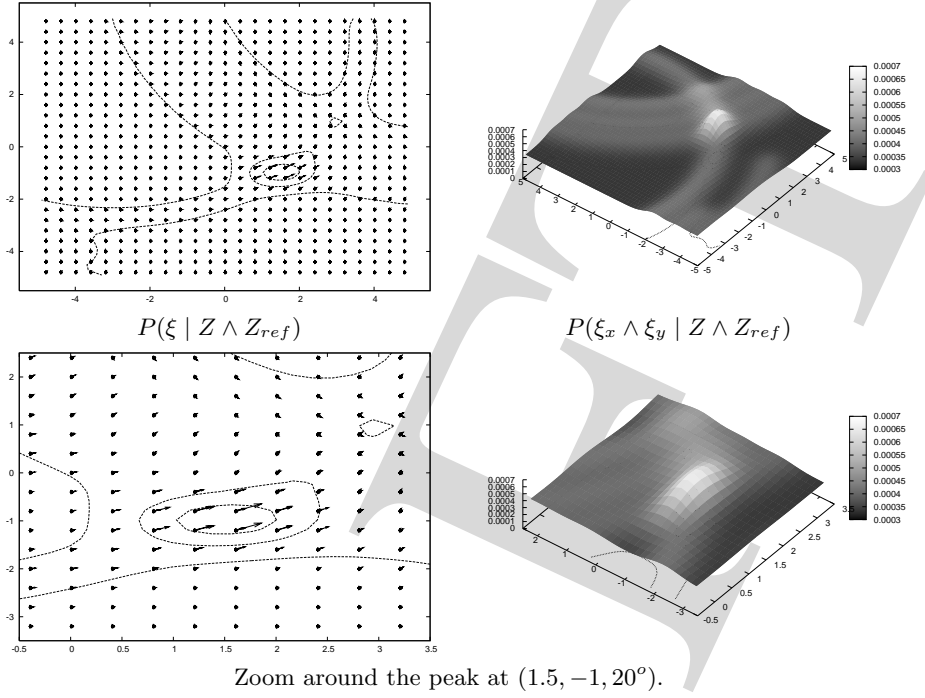


Fig. 7. Building of a probabilistic localization: diagnosis model.

Using our sensor model, we will be able to estimate the temporal position  $\tau^0$  corresponding to  $C$ . Formally, we want to compute the distribution  $P(\tau^0 | Z^0 \mathcal{T}_{sm})$ , that is, the distribution on  $\tau^0$  knowing the initial observation  $Z^0$  (made at  $C$ ) and the SMT  $\mathcal{T}_{sm}$ . To this end, we define the Bayesian program presented in Fig. 8.

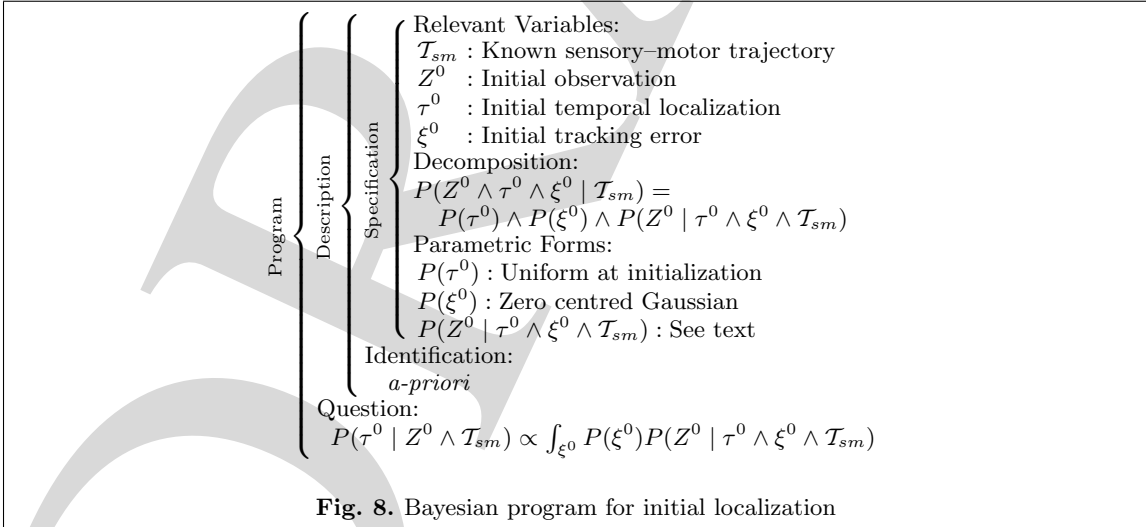


Fig. 8. Bayesian program for initial localization

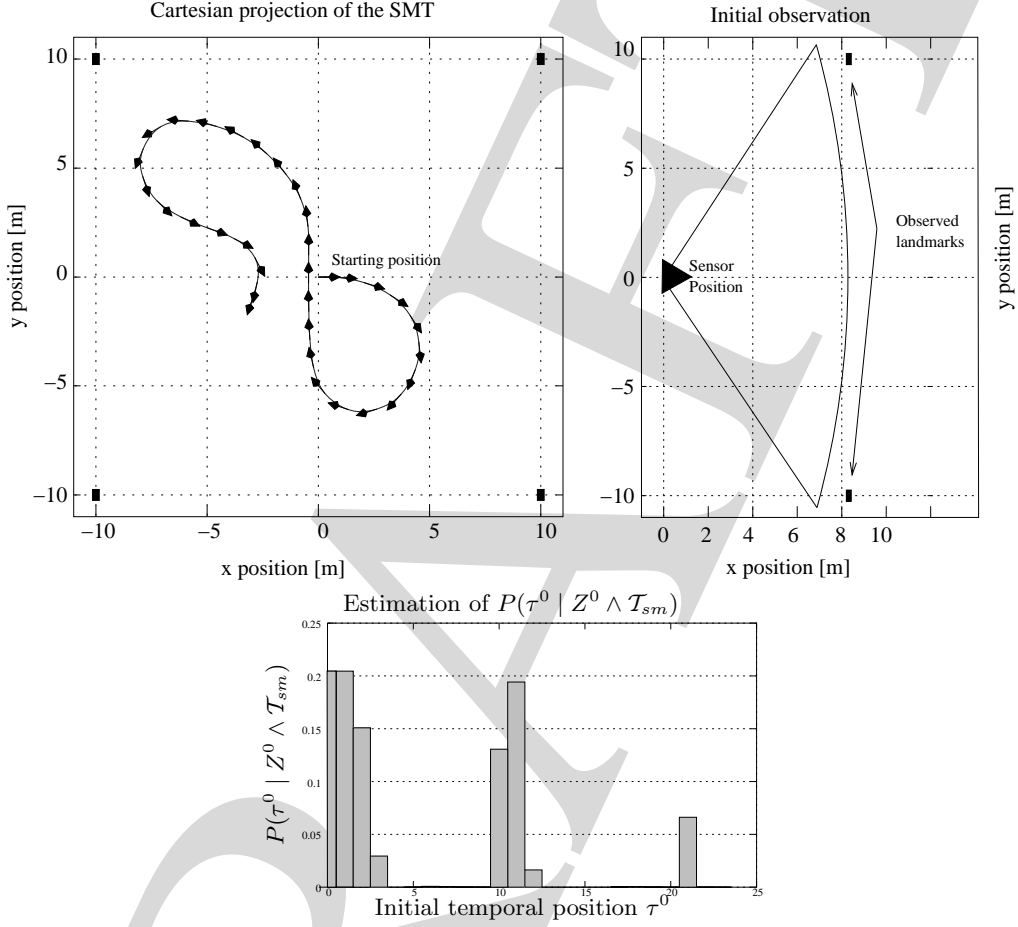
Two points must be considered in this program. First, assuming that configuration  $C$  is in the neighbourhood of the SMT implies that the tracking error  $\xi^0$  is small. To express this knowledge formally, we define  $P(\xi^0)$  as a Gaussian distribution, centred on zero. The covariance of this distribution quantifies the “neighbourhood” notion.

Second, we use the Bayesian sensor model defined in the previous sections to express the distribution  $P(Z^0 | \tau^0 \xi^0 \mathcal{T}_{sm})$ . This is achieved with  $Z_{ref}^0 = \mathcal{T}_{sm}(\tau^0).Z$  and:

$$P(Z^0 | \tau^0 \wedge \xi^0 \wedge \mathcal{T}_{sm}) = P(Z^0 | Z_{ref}^0 \wedge \xi^0). \quad (4)$$

### Estimation of $P(\tau^0 | Z^0 \wedge \mathcal{T}_{sm})$

Using the Bayesian program shown in Fig. 8, we can compute a numerical approximation of  $P(\tau^0 | Z^0 \wedge \mathcal{T}_{sm})$ . Figure 9 shows a typical result of this distribution.



**Fig. 9.** Computation of  $P(\tau^0 | Z^0 \wedge \mathcal{T}_{sm})$  for a SMT in a four-landmark environment.

From this estimated distribution (called  $\hat{P}$  below), we must extract a single value  $\hat{\tau}^0$  for  $\tau^0$ . Depending on the shape of the distribution, the complexity of this extraction can range from easy to difficult or even impossible. Several cases are possible.

1.  $\hat{P}$  is unimodal and strongly peaked. In this case, there is no ambiguity, and  $\hat{\tau}^0$  is the mode of the distribution.
2.  $\hat{P}$  is unimodal but widely spread (e.g. a Gaussian with a large standard deviation). Then, the expectation  $E[\tau^0]$  may be used as  $\hat{\tau}^0$ . This situation is often caused by a pause in the reference trajectory, resulting in several successive similar observations.
3.  $\hat{P}$  has several strong peaks. This is the consequence of some perceptual aliasing in the nominal trajectory. In this case, we must choose one of the modes. This choice may be based on some rules on the distribution (peak width, peak height, entropy,...) or from some active perception manoeuvre.
4.  $\hat{P}$  has multiple wide modes or is close to a uniform distribution. This happens when the system cannot find observations in  $\mathcal{T}_{sm}$  close to the current observation. This is probably the result of an initial position too far from the neighbourhood of  $\mathcal{T}_{sm}$ . A failure report should be generated.

## Discussion

One point should be noted here. Even when  $\hat{P}$  has a strong peak, there is no guarantee that we are indeed observing some part of  $\mathcal{T}_{sm}$ . Nevertheless, without other information, we believe that we must start moving as if we were confident with respect to our first estimation, while keeping in mind that future observations may contradict this estimation. This will be further discussed in Section 6.

### 5.1 Tracking

Once the system has computed the initial localization, it can start navigating on the SMT: it can simply track its temporal localization and its tracking error, and rely on a well-tuned controller (this will be discussed in Section 7).

The goal of the tracking is to maintain an estimate of  $\tau^t$  and  $\xi^t$  according to the sequence of observations  $\mathcal{Z}^t = Z^{0:t}$  and the sequence of controls applied to the system  $\mathcal{U}^t = U^{0:t}$ . Expressed this way, this goal is exactly the objective of a Bayesian filter.

$$P(\tau^t \wedge \xi^t \mid \mathcal{Z}^t \wedge \mathcal{U}^t) \propto P(Z^t \mid \tau^t \wedge \mathcal{T}_{sm}) \times \iint_{\xi^{t-1} \tau^{t-1}} P(\xi^{t-1} \wedge \tau^{t-1} \mid \mathcal{Z}^{t-1} \wedge \mathcal{U}^{t-1}) P(\xi^t \wedge \tau^t \mid \xi^{t-1} \wedge \tau^{t-1} \wedge U^t) d\xi d\tau$$

The practical computation of the above filter in real time is not possible for trajectories longer than a couple of seconds. Some simplifications are required:

- $\tau^t$  and  $\xi^t$  are estimated independently;
- only the most likely  $\tau^t$  is used in the computations; and
- the Bayesian filter estimating  $\xi^t$  is implemented using the well-known particle filter (also called condensation filter), assuming that  $\tau^t$  is perfectly known.

## 6 Self-confidence estimation

As shown in Section 5, our trajectory tracking system is based on a first estimation of the initial temporal position. Phenomena such as perceptual aliasing<sup>2</sup> mean that the system cannot guarantee that this initial estimation is correct. However, the robot must start moving, gathering information while following its sensory–motor trajectory. We use information gained during this movement to track a variable that expresses the confidence of the system with respect to its previous assumptions and its current localization.

Formally, variable  $\text{Conf}^t \in \{0, 1\}$  will represent system self-confidence at time  $t$ : when  $P([\text{Conf}^t = 1]) = 1$ , the system is fully confident in its localization (this should mean that it has collected a great deal of evidence); conversely,  $P([\text{Conf}^t = 1]) = 0$  expresses quasi-certainty that some failure has occurred, such as a wrong initial localization or an environment change.

In many problems where state estimation is involved, model comparison is used to *diagnose* system state (see Murphy [1998] or Lerner et al. [2000] for instances). In the remainder of this section, we will show how model comparison can be used to maintain an estimate of the system self-confidence.

### 6.1 Model comparison

#### Principle

Let us assume that we work with a variable  $A$  that can be evaluated with two distinct models. We can build the following joint distribution.

$$P(A \text{ Model}) = P(\text{Model})P(A \mid \text{Model})$$

$P(\text{Model})$  expresses our prior knowledge about *which model is the best*, and  $P(A \mid \text{Model})$  evaluates the probability distribution on  $A$  knowing which model is used. We can then use Bayes' rule to compute  $P(\text{Model} \mid [A = a])$ , i.e. given a real observation  $a$  of  $A$ , what is the model that best explains  $A = a$ ?

#### Application

Self-confidence, as defined above, can be used as a switch between two models: one that expresses which observation can be expected given full confidence in temporal localization, and one that expresses full distrust.

<sup>2</sup> Environment perceptual ambiguity.

Formally, we define the probability distribution of observation  $Z^t$  according to the system's self-confidence:  $P(Z^t | \text{Conf}^t)$ . If  $\text{Conf}^t = 0$ , because the system does not know that it knows nothing about its localization,  $P(Z^t | [\text{Conf}^t = 0])$  is a “minimal knowledge” uniform distribution, otherwise,  $P(Z^t | [\text{Conf}^t = 1])$  is set to the complete sensor model  $P(Z^t | \mathcal{T}_{sm})$  as defined in sec. 4.3.

### Tracking

Like previous tracking approaches in this chapter, self-confidence tracking will be implemented with a Markovian Bayesian filter.  $P(Z^t | \text{Conf}^t)$ , as defined above, gives us an observation model, so we simply define a transition model  $P(\text{Conf}^t | \text{Conf}^{t-1})$  to obtain:

$$P(\text{Conf}^t | Z^t) \propto P(Z^t | \text{Conf}^t) \sum_{\text{Conf}^{t-1}} P(\text{Conf}^t | \text{Conf}^{t-1}) P(\text{Conf}^{t-1}). \quad (5)$$

$$P(\text{Conf}^t | \text{Conf}^{t-1}) = \begin{array}{c|cc} & \text{Conf}^{t-1} & \\ \text{Conf}^t & 0 & 1 \\ \hline 0 & 1 - \lambda & \delta \\ \hline 1 & \lambda & 1 - \delta \end{array} \quad (6)$$

$P(\text{Conf}^t | \text{Conf}^{t-1})$  is defined by eq. 6. Parameters  $\delta$  and  $\lambda$  are called respectively the *doubting rate* and the *trusting rate*, because they express the proportion of confidence, going respectively from 1 to 0 and from 0 to 1, between two time steps.

### 6.2 Using innovation

Self-confidence tracking implemented as described above works well, except for an unwanted behaviour that appears when the robot does not move: if the current observation is well supported by the confident model, self-confidence rapidly converges to 1. This behaviour is not satisfying because the confidence changes without obtaining new information.

Because we are convinced that self-confidence should only increase when the system is able to predict a challenging observation, we designed a slightly different model that can implement the desired behaviour.

#### Observation model

Instead of using confidence as the model switch, we introduce a new switch variable  $\text{Mode} \in \{0, 1, 2\}$ . From this variable, we build an observation model  $P(Z^t | \text{Mode})$  such that:

- $P(Z^t | [\text{Mode} = 0])$  is a uniform distribution (equiv. to  $P(Z^t | [\text{Conf}^t = 0])$  in previous model);
- $P(Z^t | [\text{Mode} = 1]) = P(Z^t | Z^{t-1} \wedge U^{t-1})$  expresses the expected observation knowing only the last observation and the displacement; and
- $P(Z^t | [\text{Mode} = 2]) = P(Z^t | \tau^t \wedge \mathcal{T}_{sm})$  expresses the expected observation knowing the maximum information: sensory-motor trajectory, temporal position and tracking error distribution.

From this model and a uniform prior  $P(\text{Mode})$ , we define a joint distribution:  $P(Z^t \wedge \text{Mode}) = P(\text{Mode})P(Z^t | \text{Mode})$  from which we can compute  $P(\text{Mode} | Z^t)$  using Bayes' rule.

#### Confidence evolution model

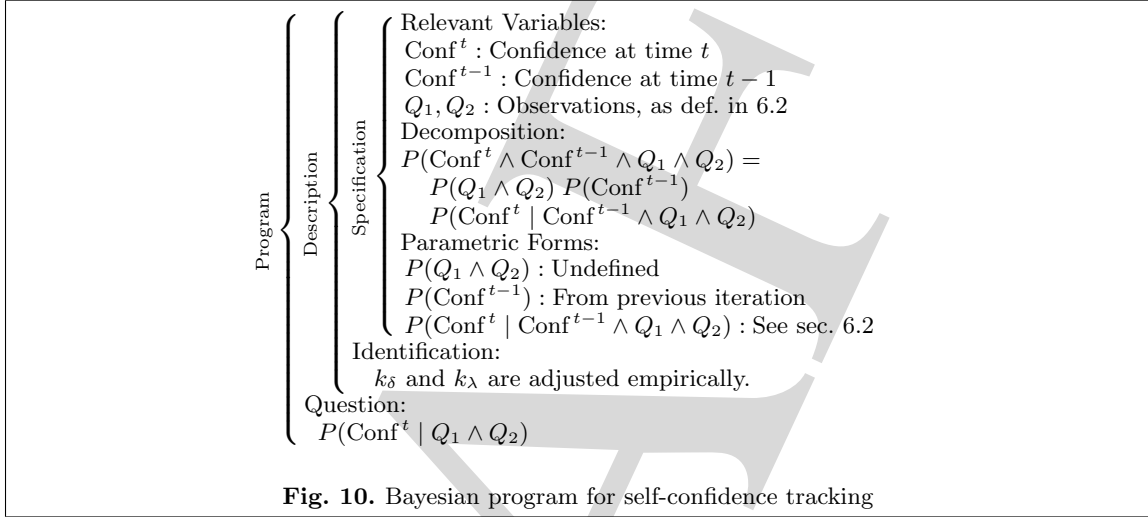
For  $i \in \{0, 1, 2\}$ , let us call  $M_i = P([\text{Mode} = i] | Z^t)$ , and  $Q_i = \frac{M_i}{M_0}$ . Knowing  $Q_1$  and  $Q_2$ , we can predict how confidence should evolve, and then design a probabilistic model that implements this behaviour.

- If  $Q_2 < 1$ , observation prediction knowing the sensory-motor trajectory and temporal position is worse than that without prior knowledge. As in Section 6.1, confidence should decrease in this case: smaller values of  $Q_2$  relative to 1 indicate bigger decreases.
- If  $Q_1 \geq Q_2 \geq 1$ , observation prediction is better knowing only the last observation than with maximum knowledge. This means that the current observation does not reflect any innovation with respect to the previous one. As there is no reason to change confidence without new evidence or counter-evidence, confidence should stay constant.
- If  $Q_2 > Q_1 \geq 1$ , the current observation not only is better predicted by the maximum knowledge observation model but also contains innovation with respect to previous observations. This ability to predict innovative observation should increase self-confidence. Furthermore, a bigger difference between  $Q_2$  and  $Q_1$  means that the current observation was very difficult to predict knowing only previous observations. Thus, larger values of  $Q_2 - Q_1$ , should increase confidence more.

To implement this behaviour, we use the Bayesian program defined in Fig. 10. This program is based on a definition of  $P(\text{Conf}^t \mid \text{Conf}^{t-1} Q_1 Q_2)$  similar to equation 6, except that the *doubting rate*  $\delta$  and *trusting rate*  $\lambda$  are now functions of  $Q_1$  and  $Q_2$ .

$$\delta(Q_1, Q_2) = \begin{cases} 0 & \text{if } Q_2 \geq 1 \\ \min(1, k_\delta(1 - Q_2)) & \text{otherwise} \end{cases} \quad (7)$$

$$\lambda(Q_1, Q_2) = \begin{cases} 0 & \text{if } Q_1 \geq Q_2 \\ \min(1, k_\lambda(Q_2 - Q_1)) & \text{otherwise} \end{cases} \quad (8)$$



An illustration of the self-confidence estimator's results will be discussed with Figs. 12, 13 and 15.

## 7 Movement

### 7.1 Trajectory following

Navigation on an SMT requires a trajectory-following controller. Knowing a temporal position estimate  $\tau^t$ , we can extract reference controls  $U_{ref}(t) = \mathcal{T}_{sm}(\tau^t).U$ . Then, with reference controls  $U_{ref}(t)$  and configuration error  $\xi^t$ , we can apply a well-tuned control law given by control theory so that the robot can accurately replay its sensory-motor trajectory.

As we wanted to design a fully Bayesian application, we used inspiration from fuzzy logic control (Klein [1999], Fraichard and Garnier [2000]) to build a probabilistic control law, expressed as a Bayesian data fusion problem:  $P(U \mid \xi \wedge U_{ref})$  being expressed as a Bayesian fusion of  $P(U \mid U_{ref})$  and  $P(U \mid \xi)$ . This controller will not be developed in this chapter, but we refer the interested reader to Pradalier et al. [2005], Pradalier et al. [2003] and Pradalier [2003] for details of the implementation.

### 7.2 Obstacle avoidance

Because our navigation system was designed to work in a moderately dynamic environment, we use controls computed for trajectory following as inputs in our obstacle avoidance module.

This module was presented in Pradalier et al. [2005] and Pradalier [2003]). Its principle is similar to such methods as *Dynamic Window* (Fox et al. [1997]) and *Ego-Kinematic Space* (Minguez et al. [2002]). Its specific advantage is mainly its expression as a Bayesian inference problem, making it particularly well suited for integration in this book's framework.

Basically, it takes as inputs data from the proximity sensors and the desired commands decided by an upper-level module (such as the trajectory following described above) and uses them to compute the commands actually applied to the robot. These commands are designed to follow the desired commands as much as possible while preserving security. We refer readers to the citations above for more details about this module.

## 8 Experimental results

### 8.1 Software architecture

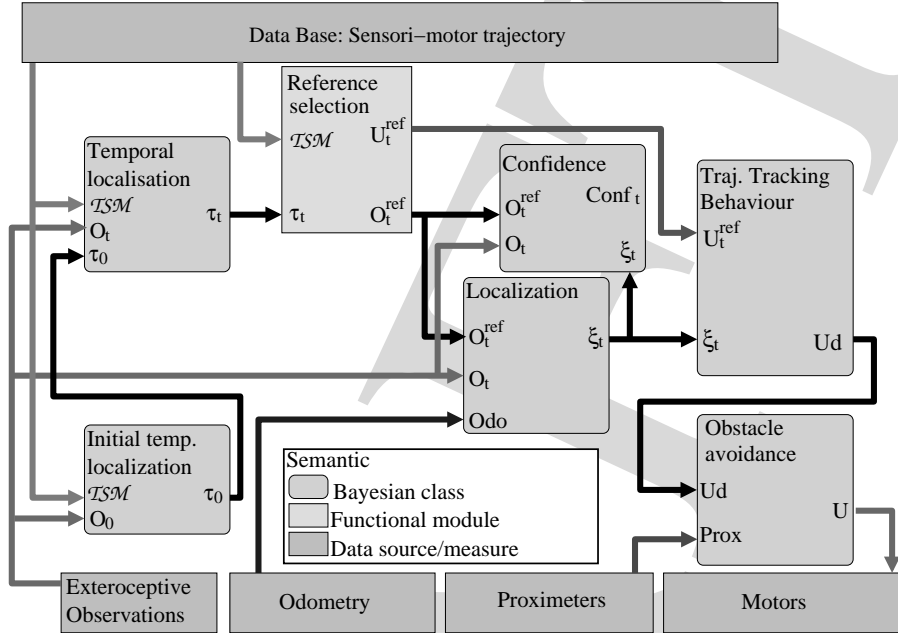


Fig. 11. Software architecture for the sensory-motor navigation application.

Our complete application architecture is presented in Fig. 11. Each Bayesian class, represented by a big rounded box, expresses a distribution  $P(\text{outputs} \mid \text{inputs})$  from which a unique value is synthesized, to reduce complexity and to achieve real-time operation. Using a multi-threaded implementation, and optimized Bayesian inference software, state tracking and controls generation can be done at about  $50Hz$ .

Notice that the output of the confidence estimation module is not connected to any other module. It is not yet clear how this information can be integrated in the control. One possibility is to make the maximum speed decrease with decreasing confidence, to make the robot more cautious when its self-confidence is low. Smarter integration of the confidence estimation within the navigation process is a subject for further research. The confidence value is currently only used as a “health status” in a human interface display.

### 8.2 Results on simulated platform

Results presented in Figures 12 and 13 are computed on a simulated CyCab, equipped with a simulated sensor, which tries to mimic as closely as possible the behaviour of the real sensor, particularly its limited field of view (180 degrees).

Both replays are made with respect to the same sensory-motor trajectory in the same environment (Figs. 12 and 13, upper left). The only difference is the initial orientation. In both situations, the initial position is a point located 1.5 m from the middle of the trajectory while the orientations differ by 180 degrees (filled triangle in Fig. 12 and 13, lower left).

From the initial perception, an initial temporal localization is computed (Figs. 12 and 13, upper right). This estimation is quasi-certain in Fig. 12, but there are more ambiguities in Fig. 13, in which the most probable  $\tau(0)$  is the one with the least unlikely match with observation.

In both cases, replay is started from the initial temporal localization, with inhibited obstacle avoidance. In Fig. 12, the initial position was quite close to the nominal trajectory (1.5 m, 0.05 radian), so trajectory tracking rapidly brings the robot to its nominal trajectory (lower left), and confidence rises to 0.9: quasi-full confidence that localization was good and replay successful. Conversely, in Fig. 13, the initial position was far from the reference ( $\pi$  radians), so the initial temporal localization was wrong. In this case, trajectory tracking tries to follow the hypothesized trajectory,

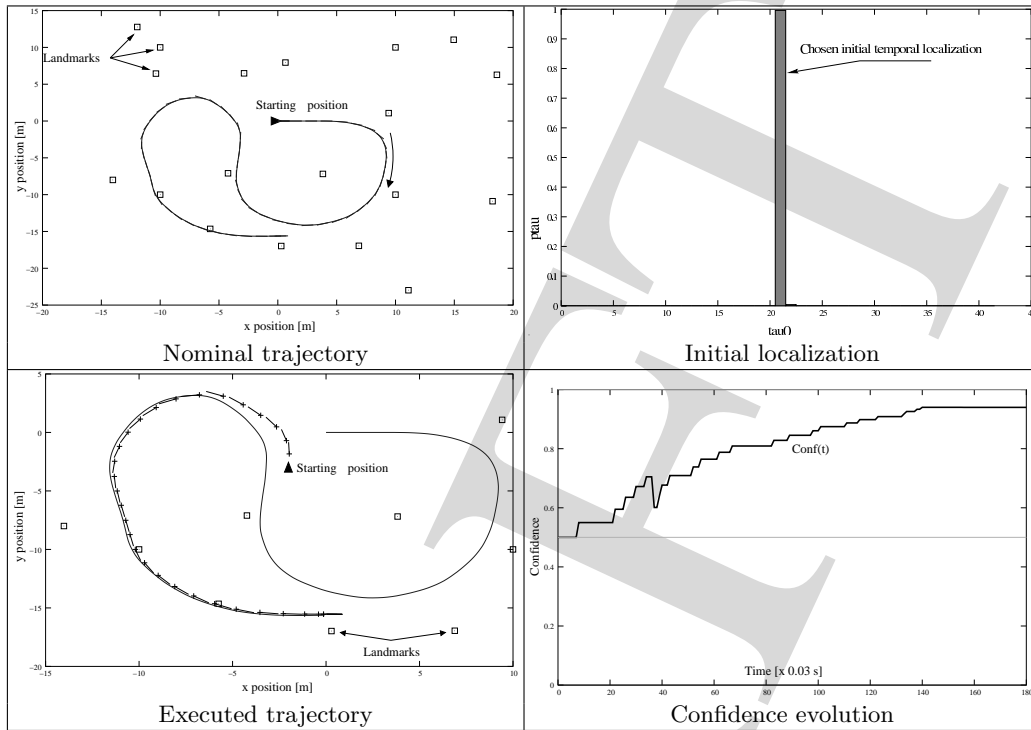


Fig. 12. Successful initialization and replay.

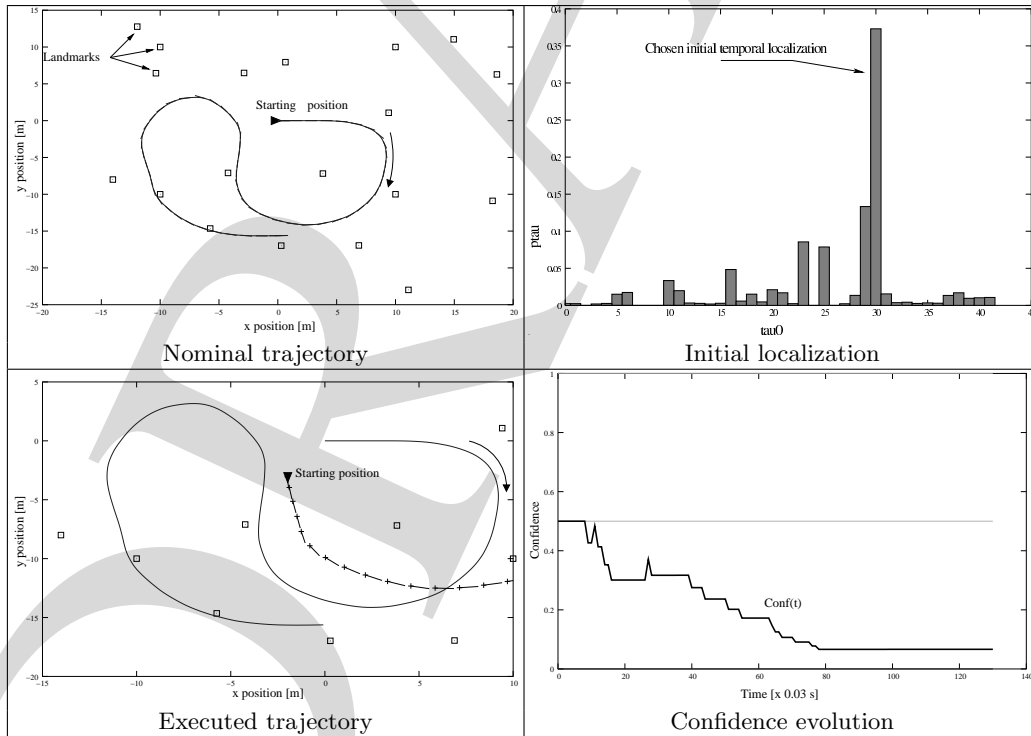
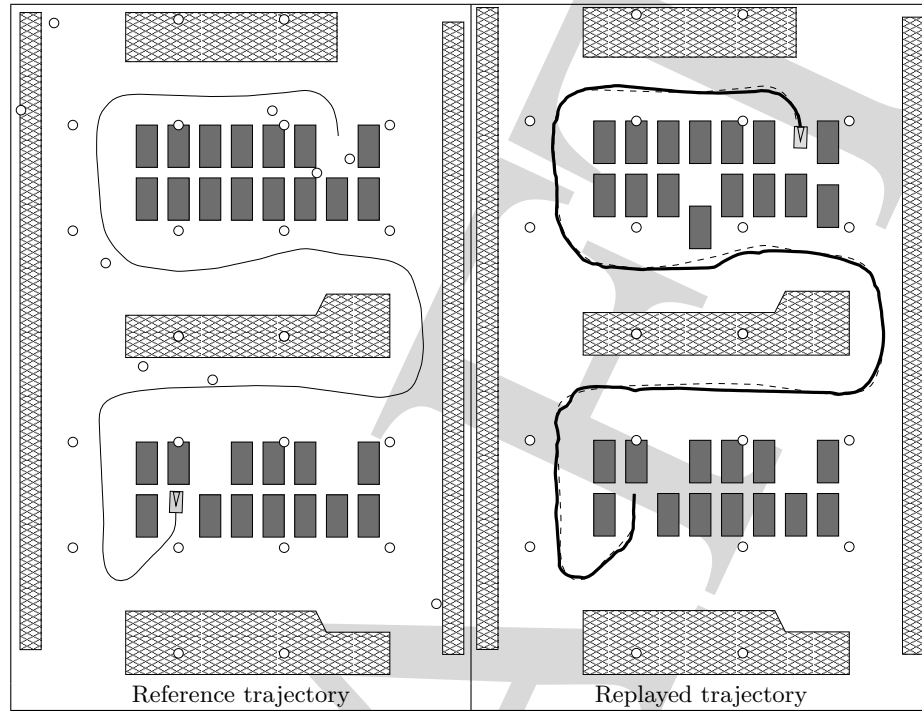


Fig. 13. Failed initialization and replay with failure diagnosis.

but evidence against the current hypothesis accumulates and confidence rapidly decreases toward zero.



**Fig. 14.** Sensory-motor trajectory tracking in a simulated parking area.

Figure 14 gives an overview of the robustness of the approach in a more realistic environment: a simulated parking area (buildings hashed, cars in dark gray, landmarks as white circles). The left-hand side of the figure displays the reference trajectory with the robot's starting position (in light gray) and the initial environment. Before starting the replay of the trajectory, we remove 10 of the 34 landmarks, and we move one of the cars to make it interfere with the reference trajectory. In the right-hand side of the figure, the reference trajectory is shown in bold black whereas the executed one is shown in dashes. Tracking is performed accurately while there is no risk of collision, even with strong curvature and missing landmarks. Deformations of the trajectory occurred around the moved car and when the reference trajectory was too close to buildings.

### 8.3 Results on real platform

Figures 15 and 16 illustrate the results of the replayed trajectory on our real car-like robot. Our landmark detector is used as sensory input, and obstacle avoidance is used to check whether the proposed controls are safe. Trajectory replay is executed accurately at 2 m/s, moving at a few tenths of a centimetre from parked cars.

The crossing of a pedestrian, in Fig. 16, is handled gracefully by the application: the trajectory is deformed until the only safe action is to give way to the pedestrian.

## 9 Conclusions

We have shown how behavioural navigation on a sensory-motor trajectory can be expressed as a fully Bayesian application: temporal and spatial localization, control generation, obstacle avoidance and failure diagnosis were successfully implemented and integrated on a simulated robot and on a car-like autonomous vehicle in the car park of our institute.

It is important to note that we used probabilistic reasoning to design a behaviour that is computationally efficient, predictable and safe for the vehicle and its environment.

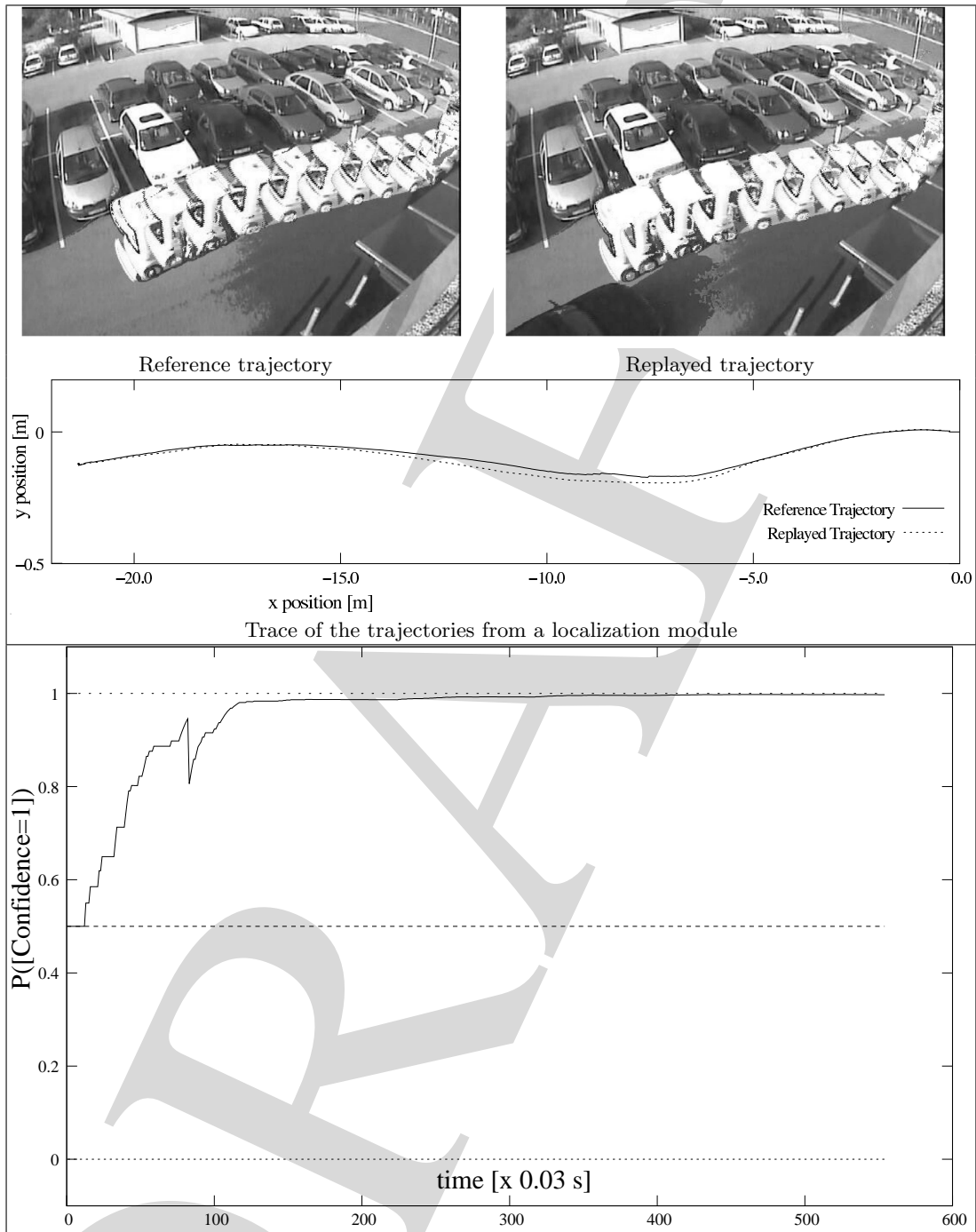


Fig. 15. Sensory-motor replay on a car-like vehicle: experiment 1.



**Fig. 16.** Sensory-motor replay on a car-like vehicle: experiment 2.

DRAFT

## References

- R. Arkin. Reactive robotic systems. *The handbook of brain theory and neural networks*, pages 793–796, 1998.
- R. P. Bonasso, D. Kortenkamp, D. Miller, and M. Slack. Experiences with an architecture for intelligent reactive agents. In *Proc. of the Int. Joint Conf. on Artificial Intelligence*, Montreal (CA), Aug. 1995.
- F. Chaumette. Visual servoing using image features defined upon geometrical primitives. In *Proc. of the Int. Conf. on Decision and Control*, Lake Buena Vista, FL (US), Dec. 1994.
- D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33, Mar. 1997.
- T. Fraichard and P. Garnier. Fuzzy control to drive car-like vehicles. *Robotics and Autonomous Systems*, 34(1):1–22, Dec. 2000.
- J. Hermosillo, C. Pradalier, S. Sekhavat, and C. Laugier. Experimental issues from map building to trajectory execution for a bi-steerable car. In *Proc. of the IEEE Int. Conf. on Advanced Robotics*, Coimbra (PT), July 2003a.
- J. Hermosillo, C. Pradalier, S. Sekhavat, C. Laugier, and G. Baille. Towards motion autonomy of a bi-steerable car: Experimental issues from map-building to trajectory execution. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Taipei (TW), May 2003b.
- L. A. Klein. *Sensor and Data Fusion Concepts and Applications*. Society of Photo-Optical Instrumentation Engineers (SPIE), Bellingham, WA, USA, 1999.
- F. Lamiroux, S. Sekhavat, and J.-P. Laumond. Motion planning and control for hilare pulling a trailer. *IEEE Trans. Robotics and Automation*, 1999.
- J.-P. Laumond, T. Simon, R. Chatila, and G. Giralt. Trajectory planning and motion control for mobile robots. In J.-D. Boissonnat and J.-P. Laumond, editors, *Geometry and Robotics*, volume 391 of *Lecture Notes in Computer Science*, pages 133–149. Springer, 1989.
- U. Lerner, R. Parr, D. Koller, and G. Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Proc. of the Nat. Conf. on Artificial Intelligence*, Austin, TX (US), Aug. 2000.
- E. Malis, G. Morel, and F. Chaumette. Robot control using disparate multiple sensors. *Int. Journal of Robotics Research*, 20(5):364–377, 2001.
- J. Minguez, L. Montano, and J. Santos-Victor. Reactive navigation for non-holonomic robots using the ego kinematic space. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Washington, DC (US), May 2002.
- K. P. Murphy. Switching kalman filters. Technical report, U. C. Berkeley, 1998.
- C. Pradalier. *Intentional Navigation of a mobile robot*. Thèse de doctorat, Inst. Nat. Polytechnique de Grenoble, Dec. 2003.
- C. Pradalier, J. Hermosillo, C. Koike, C. Braillon, P. Bessière, and C. Laugier. Safe and autonomous navigation for a car-like robot among pedestrian. In *IARP Int. Workshop on Service, Assistive and Personal Robots*, Madrid (ES), Oct. 2003.
- C. Pradalier, J. Hermosillo, C. Koike, C. Braillon, P. Bessière, and C. Laugier. The cycab: a car-like robot navigating autonomously and safely among pedestrians. *Robotics and Autonomous Systems*, 50(1):51–68, 2005.

DRAFT