Evaluating the Best arrangement for Microradars at a sensored crosswalk

CSCRS Road Safety Fellowship Report

Guillaume Goujard
M.S in Systems Engineering
Civil and Environmental Engineering Department
University of California, Berkeley

March 29, 2020

Abstract

Smart intersections are increasingly being studied as a potential way to make the roads safer by tracking and warning road users from dangerous behaviours but also raising awareness from public officials on the hazard nature of certain intersection designs. Recently, a new non-intrusive solution for round-the-clock monitoring at intersection has emerged to improve road users tracking. Wireless sensors installed within a pavement transmit discrete data and fast algorithms are used to classify the different modes of trajectory. In this report, we focus on finding the best arrangement possible for sensors at an intersection. In a first part, we modeled the gain of the micro-radars at a specific intersection. Then, we proposed a detection score taking into input the diversity of pedestrian trajectories and the gain function of micro-radars. Finally, we developed a Python package available on *github* for research purpose. This package finds the best arrangement of sensors given input from the user and also simulates the outputs of in-pavement sensors given pedestrian trajectories. While we modeled specifically the gain of microradars installed at the Danville intersection in California, the purpose was mainly to develop a general framework to be applicable for different types of sensors. It can be used to simulate the output of micro-radars in function of pedestrian trajectories taken as input and to yield the best arrangement for microradars at a sensored crosswalk.

Acknowledgements

I would like to express my great appreciation to Dr. Offer Grembek for his valuable and constructive inputs during the development of this research work.

Funding for this project was provided by UC Berkeley Safe Transportation and Research Education Center (SafeTREC) and the Collaborative Sciences Center for Road Safety (CSCRS), a U.S. Department of Transportation-funded National University Transportation Center led by the University of North Carolina at Chapel Hill's Highway Safety Research Center.

Contents

1 Introduction and Problem Setting

1.1 Intersections, pedestrians and AVs

Intersections are a crucial road infrastructure where multiple trajectories of diverse modes of transportation meet and cross, this make it a hazardous place by nature. Intersections present a unique interface between multiple and complex trajectories for different modes of transportation. Indeed, different modes are changing lines and interacting with each others. These modes can be cars, bikes, pedestrians for the most obvious but -increasingly- new modes of transportation called micro-mobility solutions (e.g. electrified bikes, scooters, skateboards) are raising some new challenges. These actors present difficult behaviors to predict and anticipate. This explains why the majority of crash happens at intersections. Based on the Fatality Analysis Reporting System (FARS) and National Automotive Sampling System-General Estimates System (NASS-GES) data, about 40 percent of the estimated 5,811,000 crashes that occurred in the United States in 2008 were intersection-related crashes [1]. Moreover, one of the most exposed transportation mode at intersections are pedestrians [2]. According to the Federal Highway Administration (FHWA) Office of Safety more than one in five pedestrian deaths is the result of a collision with a vehicle at an intersection. Concerned with this high toll, we decided to mainly focus on crosswalk monitoring for this project.

The causes for crash hazards at an intersection are numerous and the scientific litterature is well developed about what leads to crashes. All actors lack crucial information about one another when making decisions as highlighted in [3]. There is spatial uncertainty : some vehicles or road users remain invisible until the last second, and there is temporal uncertainty : one cannot know for sure the signal phases and can become stuck at the center of an intersection or pathway when the signal switches. The very design of an intersection also plays a role in driver behaviour's uncertainty The lane marking fades at the center of an intersection leading drivers to be somewhat disoriented. In this regard, significant research has been lead to optimize the safe design of intersections [4] [**?**] [5]. To locate probable conflict zones, using safety heatmap to predict probable conflicts between cars and pedestrians has yield some significant result to improve safety at intersections [6]. With this new insight, some cities have started to redesign some intersection. The City of Berkeley redesigned in Fall 2016 the intersection at Hopkins Street and The Alameda, including four concrete "refuge islands" to protect pedestrians and bikers and force vehicles to slow down.

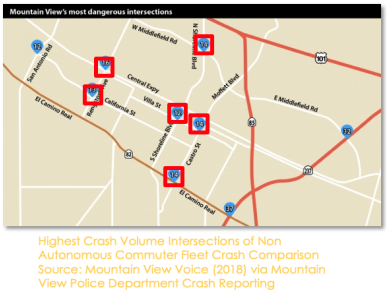


Figure 1: Highest Crash Volume for Non AV in Mountain View

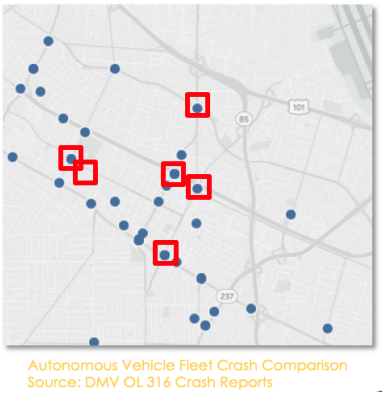


Figure 2: Crash Location for AVs in Mountain View

Apart from design, car automation is very promising to improve intersection safety. Actually, a benefit of automation raised by the National Highway Traffic Safety Administration (NHTSA) is that Automated vehicles could potentially save lives and reduce injuries since 94% of serious crashes are due to human error. The AVs could uptake the challenges of the temporal and spatial uncertainty with their in-board tech, pre-computed street maps, lidar sensors and computation power. However, up until now, the claim has not materialized into reality : the rate of "disengagement" of AV stays relatively high, worse, according to the California DMV, between October 2014 and April 2018, 58 out of 66 self-reported AV crashes in California have occurred near intersections. The two figures (1, 2), show that, where humans are faulty, the same goes for AV.

1.2 Towards an emergence of I2V

One problem comes from the fact that an AV disposes of more or less of the same information as a human driver. It cannot leverage information that could store and broadcast a smart intersection. An I2V (intersection to vehicles) technology is needed and some researcher have looked into the question and proposed some design ([7]). As the penetration of connected and automated vehicles will surely increase, I2V can potentially improve safety even further. However, if this idea seems good theoretically, the communication protocol must be made safe and reliable, the technologies of detection and of information treatment still needs to be determined and standardized.

While video-based tracking provide a flexible framework to extract a lot of measures, it represents scaling-challenges as to collect, analyse and store an important amount of data. Another solution is to develop an in-pavement system like the one installed at Diablo Road and Green Valley Road, Danville, CA and instrumented with the Safety and Mobility System (SAMS) developed by Sensys Networks, Inc. This system allows for traffic signal operation, round-the-clock vehicular mobility and safety performance monitoring. In addition, a clear advantage of this passive system is that the data collection is non intrusive.

Extensive work has been done by Dr. Grembek to develop a detailed and automated assessment of multi-modal traffic safety using in-pavement sensors. Algorithms were designed and tested, yielding good result. The use of in-pavement sensors facilities for round-the-clock monitoring could be a key to the future design of safer intersections in accordance to Vision Zero plan.

Figure 3: Danville intersection

Figure 4: Microradar detection of pedestrians

Two components are keys to design the passive system introduced previously : magnetosensors to detect vehicles and microradars to detect pedestrians. Figure 5 is showing the setup of a crosswalk at the Danville intersection : 14 microradars have been installed into the pavement. To scale up this solution, we would need to reduce the deployment of microradars while maintaining the same level and precision of detection. Therefore, the question of the best arrangement for sensors to separate the trajectories of the different modes of transport (vehicles, pedestrians, bicycles) is crucial. Yet, it has not been studied in detail.

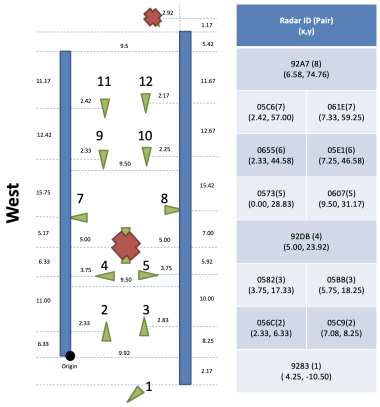


Figure 5: Microradars Setup at Pedestrian Crosswalk - Danville Intersection

1.3 Contribution of this project

The contribution of this project is double. The first contribution is to build a python package able to predict the output of the micro-radar sensors at a crosswalk with respect to a geometry of the crosswalk, the location of microradars and to a specified signal processing mechanism of a microradar. The second contribution is to lay an optimization framework to optimize the arrangement of micro-radars on a crosswalk knowing the trajectories taken by pedestrians. This project should

therefore give more insight as to where efficiently set micro-radars on a crosswalk as to optimize pedestrians monitoring.

The project is articulated as follows :

1. The first part dwells on laying a simulation framework for a single micro-radar. A simplified model is proposed and the parameters for such a model are identified using Sensys©documentation.
2. The second part presents the python package identifying modeling a sensored crosswalk. An example is developed.
3. Finally, given a distribution of possible pedestrian trajectories at a crosswalk, an algorithm and an objective function is proposed and discussed as to maximize the detection score of the crosswalk.

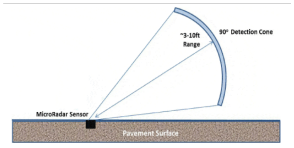
2 A model for the simulation of a single micro-radar

In this section, we care about developing a model for a single micro-radar. Given a moving an object according to a specified trajectory, we our model will output a variable named "detection bins" and accounts for the strength of the reflected RF wave. In the next section, multiple micro-radars under this model will be used obtain a model for the Danville Crosswalk. Thus, our model needs to be specific to match the characteristics of the microradars installed on the Danville Crosswalk.

2.1 Sensys Microradars

The radar developed by Sensys Networks is an in-ground radar ought to be drilled into the pavement. If its original purpose was to detect cars and bicycles, it can be calibrated to detect pedestrians as well. In principle, it works similarly to any RF radar.

(a) Installation of a sensor in the pavement



(b) Directionnality of the sensor

High frequency radio-frequency (RF) waves are transmitted in a specific direction at a fixed sample rate. When the wave meets a target object, a reflected pulse is returned and measured after being through a time-gated RF mixer. To put it more simply, the reflected pulse is scaled to account for the loss in transmission and translated to a new frequency which is named "Intermediate Frequency". A mixer is usually accompany with other eletronic components such as filters to clean out the noise and undesirable frequency [8].

Finally, the IF signal is decomposed using Fourrier decomposition and its spectrum is analyzed to obtain the presence, the distance and velocity of the object.

While we don't have the specification of the Mixer electronic circuit used by Sensys, we are going to make assumption and rely on an article to identify the parameters of a simplified model [9].

VSN240-M-2 MicroRadar Sensor	
Characteristic	Specification
frequency	6.3 GHz
radar field of view	$\pm 90°$ (azimuth), $20 - 90°$ (elevation)
maximum range	$4'(1.2m)$ to $10'(3m)$ (selectable)

2.2 Our model

The RF burst is multiplied by the Reflection in a RF mixer, the IF Signal is then decomposed and from its detection bins useful features are output and communicated to the intersection servers.

Since we do not have access to the RF mixer specifications, we propose in this report to take this first part as a black box and to simulate directly the IF Signal. In the next sections we are going to argue which parameters we need to identify and how to do so.

2.3 Parameters identification

2.3.1 IF Signal

In Sensys article [9], some graphs and datas are provided corresponding to the results of a reference detection of a Bicycle Rider. The crankshaft of the bicycle was used as the reference point for the

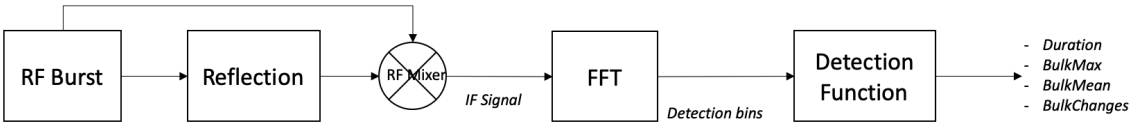


Figure 7: Theoretical model of a Sensys Microradar

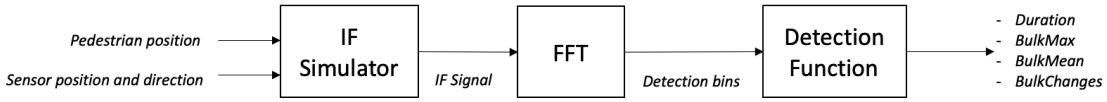


Figure 8: Simplified model of a Sensys Microradar

distances used in the graphs. We can expect to identify some parameters from these experiments. Namely there are two function to simulate : the burst and the reflection. The return signal is an attenuated and time-shifted copy of the original transmitted signal. There is also noise both in the emitted signal and reflected signal.

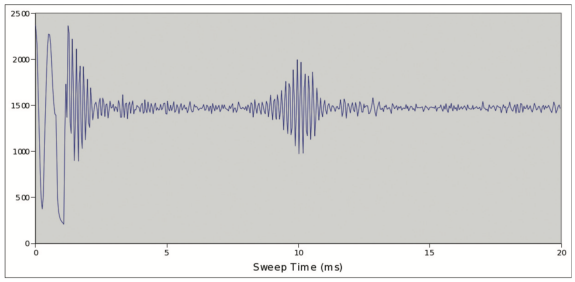


Figure 3: 512 IF Samples showing big bang and target at 6'

Figure 9: IF signal showing burst and reflection to a target located at 6'

The burst is the signal from 0 to 2 ms and is emitted to a specified rate fixed by the user. The reflection signal starts at 8 ms and ends at 12 ms and corresponds to the reflection of the burst to a target located at $6'$ from the microradar.

If we neglect the Dopler shift, the frequency of the reflection f is the same as the burst. The amplitude of the reflection $G(d, \theta)$ is given by the radar-gain function that we will estimate in the following. Finally, we can estimate the damping of the IF wave $\frac{1}{\tau}$ and its celerity to be c.
We used the functions solutions to the damped harmonic oscillator to simulate the burst and reflection :

$$\begin{cases} \text{burst}(t) = e^{-\frac{2\pi t}{\tau}} \sin(2\pi f t) \\ \text{reflection}(t,\ d, \theta) = G(d, \theta) e^{-\frac{2\pi (t - \frac{2d}{c})}{\tau}} \sin(2\pi f t) \end{cases}$$

2.3.2 Radar Gain

Now we would like to evaluate the gain function : $G(d, \theta)$.

On the right is the Radar Gain Pattern from the MicroRadar documentation. It is a graphical representations of the radiation characteristics of the antenna. For 4 targets at 4 different distance (4, 6, 8 and 10 feet), the surface represents the relative intensity of the energy radiation as a function of the angle from between direction of the microradar and the target. Our assumption is that $G(d, \theta) = A(d)B(\theta)$. The gain of a radar is defined as the ratio between the burst and the reflection, $G(d, \theta) = \frac{\max\{reflection(t,d,\theta)\}}{\max\{burst(t)\}}$ and the gain in DB is defined as $G_{db} = 20 * \log_{10}(G(d, \theta))$.

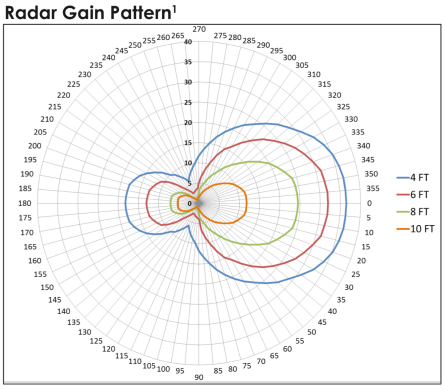


Figure 10: Radar Gain Pattern

- Identification of A.

$$A : x \to K * 10^{\frac{G_{db}(d,0)}{20}}$$

Thanks to sensys article we know that for $\theta = 0$ and 8 feet, $A(8feet) = \frac{\max\{reflection(t,8feet,0)\}}{\max\{burst(t)\}} = 0.55$ with the radar gain data, we deduce that $K = 1.410^{-2}$. Looking at the plot of the 4 points for $\theta = 0$ we choose an affine function that we try to fit an exponentially decaying gain :

$$A(x) := \alpha e^{\beta x}$$

.

- Identification of B

We only model the front lobe of the gain pattern, we decide on

$$B(\theta; \sigma) = e^{-\frac{1}{2}\frac{\theta^2}{\sigma^2}}$$

Finally our gain model for a single micro radar is :

$$G(d, \theta) = \alpha e^{\beta x} e^{-\frac{1}{2}\frac{\theta^2}{\sigma^2}}$$

From this gain model, we can now safely simulate IF signals given a position of a target, the location of the micro-radar and its angle relative to the target.

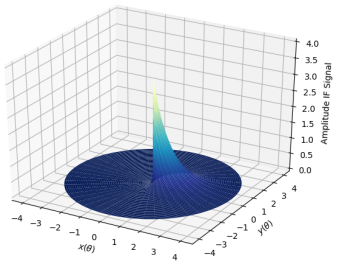


Figure 11: Modeled Radar Gain

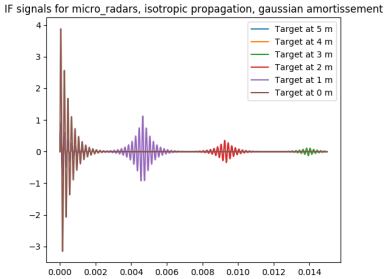


Figure 12: Different IF signals for different distance of the target

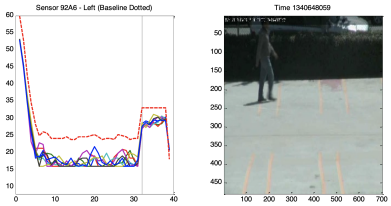


Figure 13: Microradar bins before detection : below the baseline

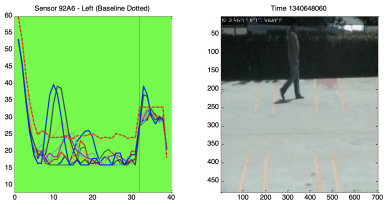


Figure 14: Microradar bins during detection : some bins are above the baseline

2.3.3 Fourier Transform

This section is an attempt to make sense out of the definition of the "detection bins" output by Sensys micro-radars. If we look at the Sensys graphs: these are two pictures of a moving pedestrian taken with a one second lapse. On the left handside of each picture, we can see the output of a process : it shows a baseline in dashed-red and 8 curves discretized by 40 bins. Each curve would be the result of the sampling process since 8 Hz is a possible sampling rate for those microradars. Without further details from the documentation, we made some extra assumption.

We argue that a Fast Fourier Transform (FFT) may be used to generate a decomposition of the signal into "bins" which could simply be the amplitudes of each frequency component. The FFT technique is a highly optimized implementation of the Discrete Fourier Transform (DFT) and is common among radars design. This technique is used for beamforming, doppler processing and pulse compression. In our case, the Sensys article states that the detection bins are defined as the number of bins above the baseline. We tried therefore to set up a tentative FFT model.

We fit the baseline over an IF signal with no reflection and then we simulate 40 bins of DFT for a sampling frequency of 67Hz.

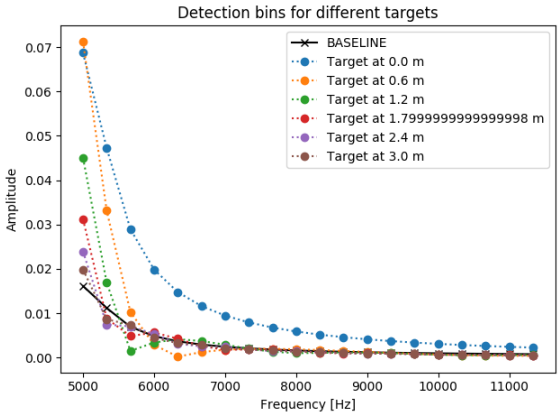


Figure 15: Detection bins for different targets

2.3.4 Detection bins or Radar Gain ?

We recall that our problem is to find the best arrangement for microradars at a crosswalk. A good sensor is a sensor that yields a strong number of detection bins when a target passes in its vicinity. However, we have seen that our way to simulate those detection bins remains experimental and is computationally heavy. At the same time, the gain function of the radar has a very simple closed form expression and we argue that maximizing its overall gain over a pedestrian trajectory is somewhat equivalent to maximizing the detection bins of a sensor over a pedestrian trajectory. If we look at plot (16) and (17), we see that "globally" the detection bins are increasing with the gain and similarly the L1 norm of the detection bins are also increasing with the gain. Therefore, in the rest of the report, we will directly work with the radar gain function instead of the detection bins. We think that further data analysis should be done to confirm or infirm our Radar model.

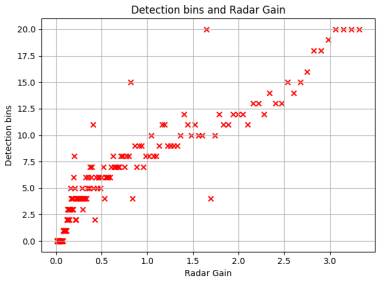


Figure 16: Detection bins in function of the gain

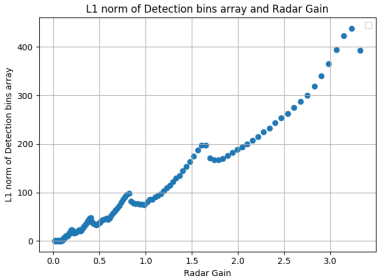


Figure 17: L1 norm of Detection bins in function of the gain

3 Optimization and Best Arrangement

In this section, we aim at setting up a problem of optimization to find the best arrangement of micro-radars at a smart crosswalk. Given frequent pedestrian trajectories on the crosswalk and a constraint on the number of micro-radars that we can install, we would like to obtain the arrangement that maximize the detection rate of the pedestrians along the trajectories. Instead of using the detection bins which is the result of a complex process as described in section 1, we use the gain function that we obtained in closed form in (2.3.2).

3.1 Crosswalk set-up

On the right hand side, figure (19), is the basic set up of a crosswalk represented as a rectangle of length l and width L. The Cartesian reference frame point at origin is located at the bottom left of the crosswalk/rectangle. We also displayed a microradar represented by an angle with its bisector. The three parameters defining a microradar are its location (x_1, y_1), and its angle relative to the y-axis θ_1. Note that in the rest of the section, a trajectory and a micro-radar can be placed outside the cross walk.

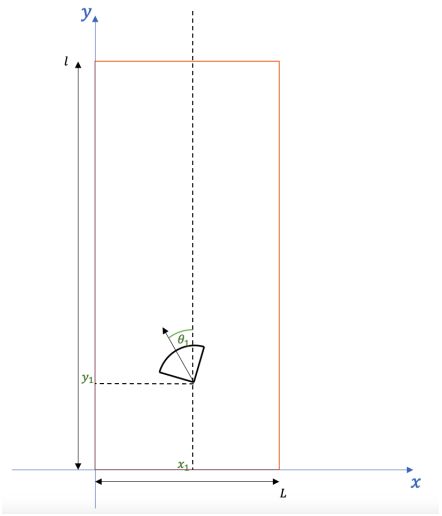


Figure 18: Geometric set up of the crosswalk

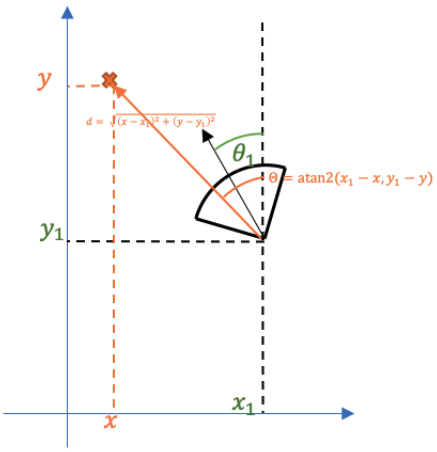


Figure 19: distance and angle relative to the sensor

Each pedestrian location is noted as (x, y), it is important to explicit its distance to the sensor and its angle relative to the y-axis. If (x_1, y_1, θ_1) is the location and angle of the sensor :

$$d = \sqrt{(x - x_1)^2 + (y - y_1)^2}$$

Similarly, for the angle the orientation of the sensor we use the atan2 function that is used to compute the principal value of the argument function applied to the complex number $(y_1 - y) + i(x_1 - x)$.:

$$\theta = \text{atan2}\big(x_1 - x, y - y_1\big)$$

Note that the angle are accounted positively from the y-axis, it is then as if we were to rotate the usual reference frame from $\frac{\pi}{2}$.

3.2 Problem Formulation

3.2.1 Sensors

Let k be the numbers of sensors that we want to install and (x_i, y_i, θ_i), their corresponding location and angle with respect to the y axis of the crosswalk. The following set contains all the information relative to the sensors :

$$\mathcal{S}_k = \{(x_i, y_i, \theta_i), i \in (1, k), (x_i, y_i, \theta_i) \in \mathbb{R}^3\}$$

The set $\mathcal{S}_k$ contains our optimization variables.

3.2.2 Trajectories

A trajectory is an affine function on the plane $f(x, y)$ that represents the trajectory of a pedestrian and an associated probability ω_i which is the frequency of usage of this trajectory. For instance, the following trajectory function $f(x, y) = y - \frac{l}{L}x$ encode for the diagonal of the pathway. All the element of the trajectories are : $\{(x, y),\ f(x, y) = 0\} = f^{-1}(0)$.
We denote as $\mathcal{T}_p$, a set of p trajectories :

$$\mathcal{T}_p = \{(\omega_i, f_i), i \in (1, p), \sum_{i=1}^{p} \omega_i = 1\}$$

3.2.3 Objective Function : the Detection Score

We want to maximize the detection rate for the pedestrians on the crosswalk. To do this, we want to maximize the gain of all the sensors along the trajectories. If we simply sum up all the gain of the sensors, our optimization will yield the same location and orientation for all the sensors. In this regard, we define the "Detection Score", a function describing how well the trajectories are detected. Under this score, two sensors in the same location won't increase the global detection score. Thinking about it, the detection of a pedestrian at a location $z = (x, y)$ is fully captured by the maximum of the gain of all the sensors, i.e, we don't mind of being detected by all the sensors when we are correctly detected by one. We propose the following detection score function D. Let $T \in \mathcal{T}_p$ and $S \in \mathcal{S}_k$, T is a fixed set of k trajectories and S a fixed set of k sensors.

$$D(S; T) = \sum_{i=1}^{p} \omega_i \oint_{f_i=0} \max_{1 \leq j \leq k} \big(g(z; x_j, y_j, \theta_j)\big) dz$$

To put it in simple words, for each trajectory and for each step of the pedestrian on this trajectory, we add the maximum gain of the sensors to the detection score. Finally, we weight the contribution to the Detection Score.
So that the detection score is smooth and differential in every of x_j, y_j, θ_j, instead of using the maximum function, we use the $\alpha - softmax$ function. This soft maximum is frequently applied in optimization and neural computation [10].

$$\mathcal{S}_\alpha(x) = \frac{\sum_{i=1}^{n} x_i e^{\alpha x_i}}{\sum_{i=1}^{n} e^{\alpha x_i}}$$

In addition, reminding that $g(z; x_j, y_j, \theta_j) = \alpha \mathrm{e}^{\beta ||z - (x_j, y_j)||_2)} e^{-\frac{1}{2} \frac{\theta(z, x_j, y_j, \theta_j)^2}{\sigma^2}}$, instead of maximizing g along the trajectories, we will maximize $\log(g)$, which changes the Detection function but will make the computation of the Jacobian faster. Let us define $\log D(T, S)$ which is a differential and smooth function in S.

$$\log D(S; T) = \sum_{i=1}^{p} \omega_i \oint_{f_i=0} \mathcal{S}_\alpha\big(\log g(z; S)\big) dz$$

where $\log(g)(z; S) = [\log(g(z; x_1, y_1, \theta_1)) \ \ldots \ \log(g(z; x_k, y_k, \theta_k)]$

3.2.4 Optimization Problem

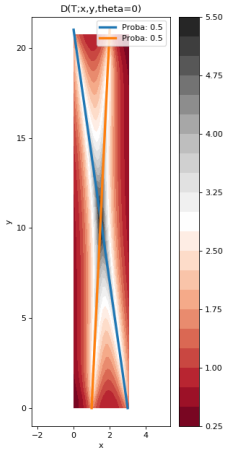
Having fixed $T \in \mathcal{T}_p$, the set of frequent trajectories on the crosswalk. Finally we derive the following stochastic optimization problem,

$$\begin{array}{ll} \min & -\log D(S;T) \\ \text{subject to} & S \in \mathcal{S}_k, \end{array} \tag{1}$$

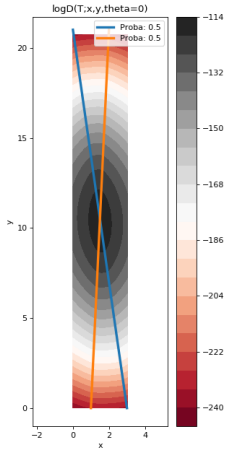
We aim at maximizing the detection score for a suitable set of locations and angle of our sensors set. In appendix (C), we figure out a simple case in closed form. If we have one sensor S to place and one trajectory T, minimizing $-logD$ is going to yield an optimal sensor on the trajectory with the angle oriented along the trajectory curve.

3.2.5 Comparision of the two detection scores

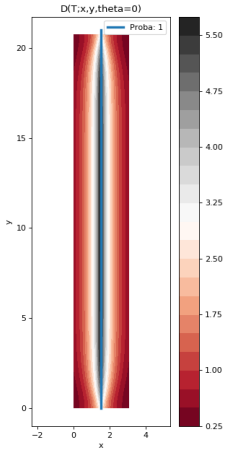
All the following plots have been simulated for a single Sensor placed in (x, y) with an angle of 0 relative the vertical axis. The first row corresponds to a set of two trajectories and the second row to a set of a single one trajectory. The first column is obtained by using the Detection Score D and the second column is obtained by using the Log Detection Score $\log D$. We see that the curve is way sharper for the real detection score than for the log detection score. We will compare the solution to the log detection to the detection score in the following sections.



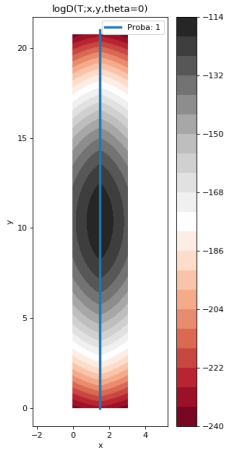
(a) $D(x, y, 0)$ for a double trajectory



(b) $\log D(x, y, 0)$ for a double trajectory



(c) $D(x, y, 0)$ for a single trajectory



(d) $logD(x, y, 0)$ for a single trajectory

Figure 20: 2 Examples of trajectories for the two detection scores.

3.3 Algorithms

Now that we have formulated a differential although non-convex objective functions and we need to propose an algorithm to solve it. We are going to integrate the trajectories over $y \in (0, l)$ and over this segment $x \in (0, L)$. Therefore although, we can make the assumption that our problem is unbounded and verify a posteriori that it is. If $\theta \notin (0, 2\pi)$, then we will take $\tilde{\theta}^* = \theta^* \mod 2\pi$. Therefore our program can be written under the more general form :

$$\min_{x \in \mathbb{R}^{3k}} \quad f(x) \tag{2}$$

The bulk of the literature for solving (2) has been focusing on the case where f is convex (e.g. [11]). The methods can be broadly classified into two types of algorithm : first orders and second-orders algorithm ([12]). If the first orders are based solely on the gradient of f, they are generally beaten in convergence property by second orders which incorporate curvature information via the Hessian. The most famous of those methods is the Newton algorithm :

$$x_{k+1} = x_k + \alpha_k p_k \qquad p_k = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

Where $x_k, \alpha_k, \nabla f(x_k), \nabla^2 f(x_k)$ are respectively the current iterate, the step-size, the gradient and the Hessian matrix ([13]). Other method tries to derive the next step p_k using a conjugate gradient. We call them Newton CG. In our case, these algorithms (Newton, Newton CG) cannot be applied, even to find a local minimum since we have no guarantee that $\nabla^2 f(x_k)$ is invertible.

Quasi-Newton methods assume that we can calculate F(x) and $\nabla F(x)$. An invertible matrix H_k approaching the inverse of the hessian is computed at each iteration and a line-search is performed on the expected decreasing direction : $-H_k^{-1} \nabla f(x_k)$, to yield the best step α_k possible. These methods are very attractive options to Newton's method in that they do not require computation of the Hessian and yet still attain an efficient rate of convergence.

$$x_{k+1} = x_k + \alpha_k p_k \qquad p_k = -H_k^{-1} \nabla f(x_k)$$

3.4 BFGS algorithm

One of the most popular quasi-method newton algorithm is the BFGS algorithm named after its inventors, Broyden, Fletcher, Goldfarb, and Shanno. We adapt the algorithm from the following book ([14]). We aim to find $S \in \mathbb{R}^{3k}$ to minimize $f(S)$

Defining $y_k = \nabla f(S_{k+1}) - \nabla f(S_k)$ and $s_k = S_{k+1} - S_k$,

Each approximation of the inverse of the hessian H_{k+1} is given by :

$$H_{k+1} = (I - \frac{s_k y_k^T}{s_k^T y_k}) H_k (I - \frac{y_k s_k^T}{s_k^T y_k}) + \frac{s_k s_k^t}{s_k^T y_k}$$

We are going to sketch roughly the algorithm. From a starting point $S_0 \in \mathbb{R}^{3k}$, a convergence tolerance and a first inverse Hessian approximation H_0 (by default the identity), while the norm of the gradient of f taken at S_k is greater than the convergence tolerance :

- We compute the search direction
$$p_k = -H_k \nabla f_k$$
- We update S_k, where α_k is found from a line search procedure,
$$S_{k+1} = S_k + \alpha_k p_k$$
- We compute H_{k+1}

3.5 Initial conditions for k sensors

In a Quasi-Newton approach we must first define the initial conditions of the sensors arrangement on the crosswalk. Knowing we have a rectangle we would be tempted to subdivise this rectangle into k sub-rectangle of the same area. But what if k is odd ? Using triangles to pave the rectangle is not an option either, we know from [15] that it is impossible to trisect a rectangle with three triangles of the same area.

Simple initialization For now we propose to dispose the k micro-radar on the axis $x = \frac{L}{2}$ and with an orientation going gradually from 0 to $-\pi$. This is what we call the simple initialization.

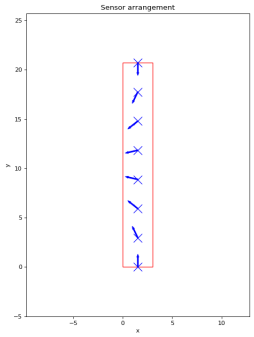
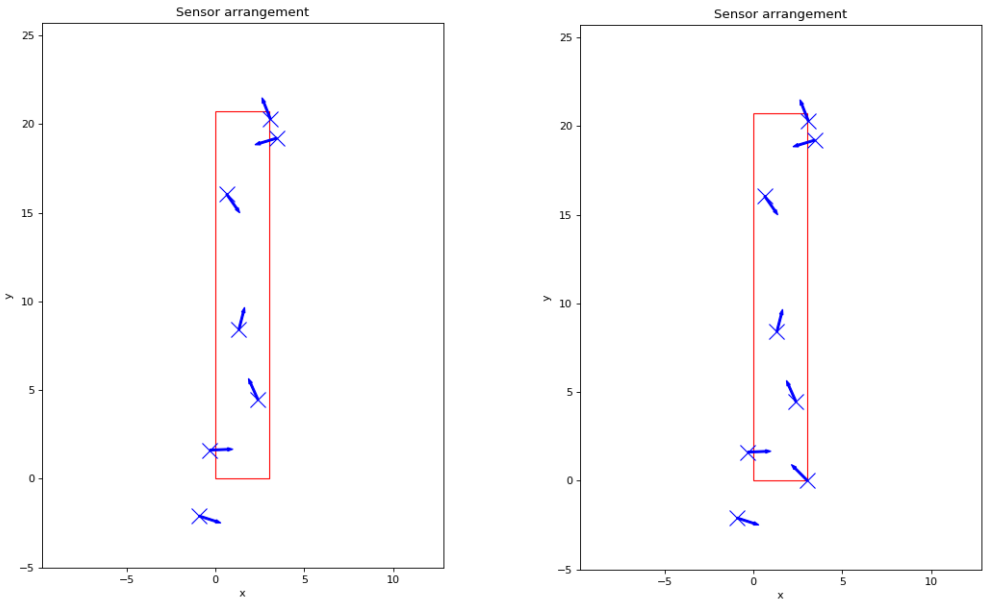


Figure 21: Simple initialization arrangement for k=8

Heuristic initialization Another idea would be to benefit from the computation of the solution for $k-1$ that we call $\hat{S}_{k-1}$ and initialize the algorithm for k sensors with the k-1 first sensors being the $\hat{S}_{k-1}$ and the last one being placed at one of the corner of the crosswalk. In the example (fig (22)) below, we add a new sensor on bottom right of the crosswalk before launching the BFGS algorithm.



(a) $\hat{S}_7$, final arrangement for k=7

(b) Heuristic Initialization k=8

Figure 22

3.6 Set of considered trajectories

For the results, we considered two trajectories crossing the crosswalk diagonally with equiprobability. We consider that those two trajectories add enough heterogeneity to provide a perspective on the results of this optimization framework.

3.7 Results

Benchmarking our algorithm is not an easy task since there is no ground truth on the best arrangement to take for sensors at a crosswalk. Ultimately, we want to compare the arrangement that our algorithm yields with the one chosen at the Danville intersection. To compare, we use the detection function D to decide on whether an arrangement is better or worse than another. We have access to the sensors used at the Danville intersection. We will not consider sensors 7 and 8 as they are used to detect cars. Finally, there are 12 sensors $\tilde{S}$. For our set of trajectories, the logD and D is :

$$-logD(T, \tilde{S}) = 168.95$$
$$D(T, \tilde{S}) = 10.22$$

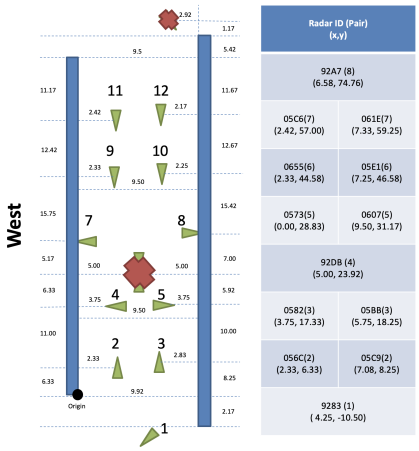


Figure 23: Danville crosswalk sensors setup

Of course, one must note that our arrangement won't be necessarily optimal with respect to the log detection function since we are using an approximation minimization algorithm on a non-convex function. We use the following set up :

- The simulated crosswalk has the same dimensions than the Danville crosswalk : 10 feet X 68 feet,
- The pedestrians are only taking two trajectories with equiprobability (see (3.6)).
- We use two different initializations (see (3.5)).
- For every number of sensors between 1 and 12 (the number of sensors on the Danville crosswalk), we are going to find the approximate solution $\hat{S}_k$ for the simple and heuristic initialization.

3.7.1 BFGS - logD and D

For the two initializations and for a number of sensors ranging from 1 to 12, we plotted : $-logD(T, S)$ for $S \in \{\hat{S}_k, \forall k \in (1, 12)\}$, and $D(T, S)$ for $S \in \{\hat{S}_k, \forall k \in (1, 12)\}$. It is worth noting the minimum number of sensors such that our optimized arrangement is better than the Danville arrangement.

As we would expect, adding more sensors is going to increase the detection score with a decreasing marginal detection score. I.e the first positioned sensor is going to have a bigger impact on $-logD$ than the next one. It is also relieving to see that as logD increases, so is the detection score D. As soon as 3 sensors are added to the arrangement, the logD score of our arrangement beat the Danville arrangement. For $k = 8$, our arrangement with the Heuristic arrangement is already better than the Danville arrangement. From those two plots we also note that starting from 6 sensors on the crosswalk, the arrangement obtained from the Heuristic initialization yields better results than the one from the simple initialization (even if the logD score is the same).

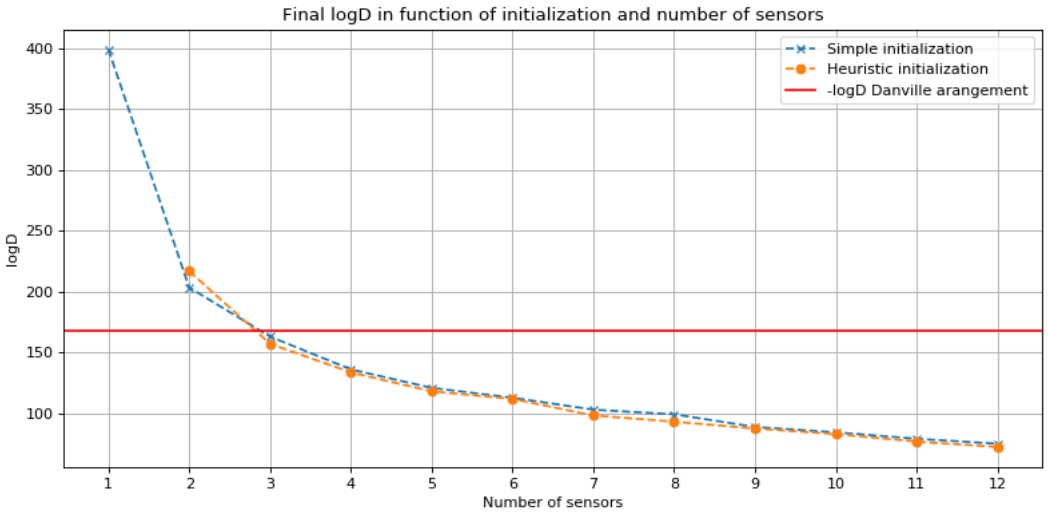


Figure 24: $-logD(\hat{S}_k)$

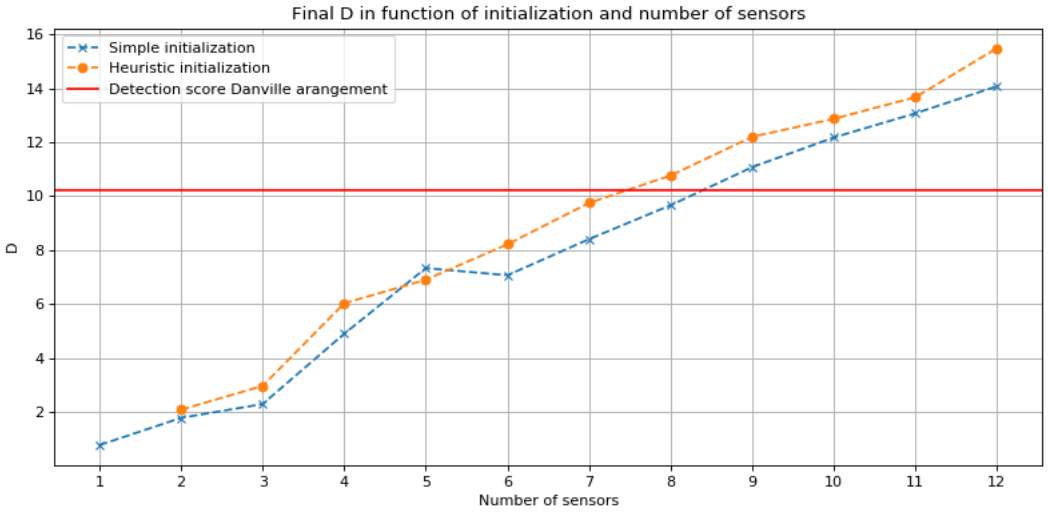
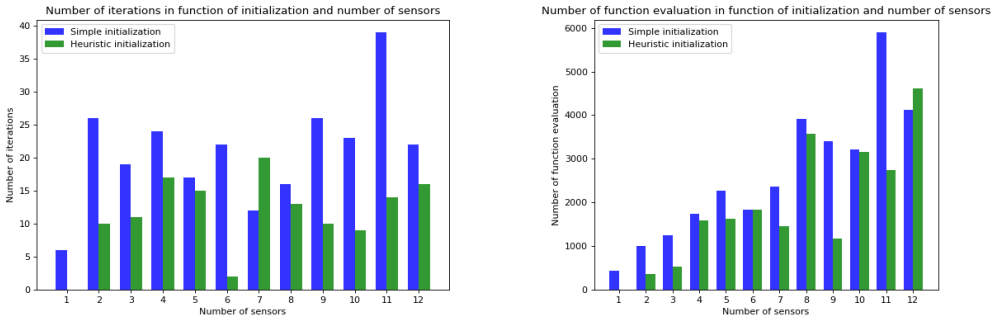


Figure 25: $D(\hat{S}_k)$

3.7.2 BFGS - Computation

Having fixed the tolerance on the norm of the Jacobian to be $1e-1$, we note that the iterations until convergence and the number of time $-logD$ is evaluated increases with k, making the computation even more expensive. It makes sense, since the dimensions of the solution space is increasing by three for each added sensor. Finally, we can notice that the Heuristic initialization generally leads to a smaller number of iterations of the BFGS algorithm and a generally smaller number of function evaluations.



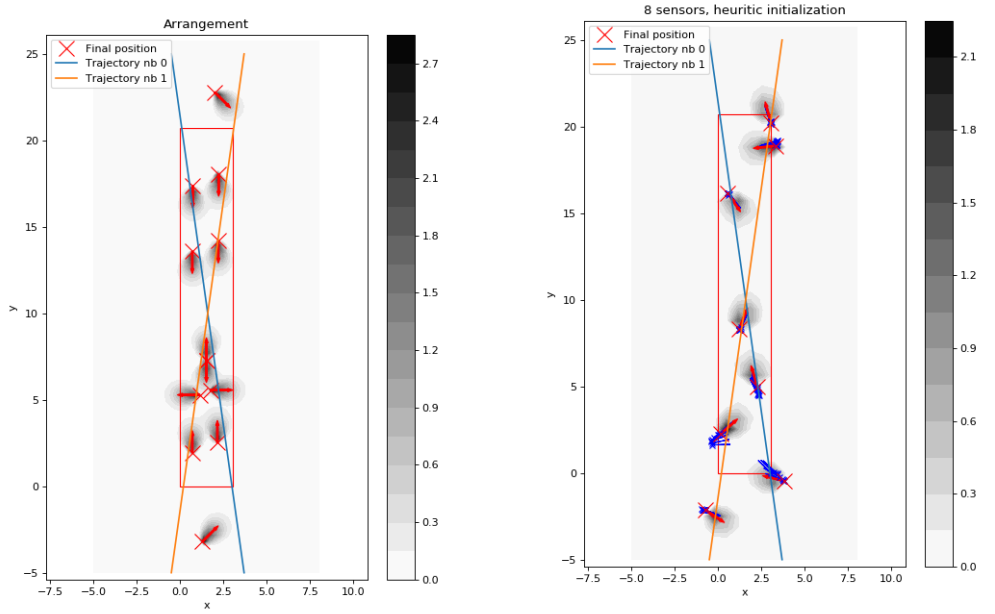
(a) Number of iterations

(b) Number of function evaluations

Figure 26

3.7.3 BFGS - Arrangements

All the arrangements for the two initializations can be seen in the appendices. However, we display our best arrangement for k=8 vs the Danville arrangement. On the left handside (figure (27a) is the arrangement currently in place at the Danville crosswalk, on the right handside (figure (27b)) is the result of our optimization process for k=8 sensors and with a Heuristic initialization. According to the detection scores (see figure (25)), the right handside arrangement yields a better detection score than the left-handside. The red arrows are the final positions and angle of the sensors, the blue arrows are the results of each iteration of the BFGS algorithm. The two lines are the trajectories considered, the red rectangle is the crosswalk. Finally, the black scale represents the gain function for each radar.



(a) Sensys arrangement

(b) Result of the optimization for k=8 sensors

Figure 27

3.7.4 Discussion

Under a reasonable micro-radar gain function and a simple but yet reasonable set of trajectories, we obtained some sensible results. A micro-radar arrangement with 8 sensors as showed on figure (27b) yields a better detection score than the default arrangement currently in use at the Danville

intersection. However, we must show some caution and reserve about the results : the arrangement displayed is not necessarily the optimum optimizer of the detection score. Moreover, the radar gain function was assumed and fitted on an article data, the trajectories were invented instead of being observed on real life data. Furthermore, the detection score is a simplified view of the purpose of those micro-radars. Indeed, we have made the assumption that we are looking to detect a single pedestrian, if we are looking for redundancy in the measurements and a way to separate easily some trajectory, it would make sense to define another detection function that somewhat sum the detection function of the micro-radars along the trajectories. Other analysis should be performed on our solutions to check for its robustness : according to the trajectories that should remain uncertain (what if a pedestrian crosses outside the crosswalk) , according to the sensor that will never be placed perfectly aligned with the result of the algorithm. We could also tweak slightly the radar gain function and notice the difference in the arrangement.

We also notice that the solution is reliant on the initialization and there must exist other sensible way to define a good first guess.

4 Python Simulation Tool

In this section, we finally introduce a python package aiming at simulating the detection bins of micro-radars at a smart crosswalk with respect to a pedestrian trajectory and aiming at finding the best arrangement for micro-radars to maximize the detection score. The package named SmartCrosswalkSimulator is available at the following url : `https://github.com/GuillaumeGoujard/SmartCrosswalkProject.`

4.1 Organization of the package

At the root of the python package, there is the main file :

- **SmartCrosswalk.py**. This file contains the *SmartCrosswalk* object that we are going to introduce in this section.

Under **parameters/** folder lies :

- **parameters/microradarParameters.py** This file contains the parameters of our simplified microradar parameters, it also contains the data that we obtained from the Sensys documentation.
- **parameters/microradarTest.py**. This file contains some of the tests for the graphical representations that are presented in this report.

Under **tools/** folder lies :

- **tools/signalProcessing.py** This file contains the functions used to process and treat the IF Signal.
- **tools/signalSimulation.py** This file contains functions used to simulate the radar gain function, IF Signal.
- **tools/scoringFunctions.py** This file contains the functions used to generate the Detection score and to optimize the arrangement of the sensors.
- **tools/optimizationCallBack.py** This file contains the class MyCallBack that is used to store the output of the optimization function "scipy.minimize". It is also used to plot the optimization results.
- **tools/plottingTools.py** This file contains the functions used to generate some plots of the crosswalk and of the simulation results.

4.2 The "SmartCrosswalk" object

This class was coded as to be easy to understand and use. A SmartCrosswalk object takes in input :

- A list of sensors
- The dimensions (width, length) of the crosswalk.
- An boolean argument "feet", if the dimensions are in feet.

4.3 Example 1 : Simulation of a trajectory

The examples presented below can be run in the main of the SmartCrosswalk.py
Let's say we want to simulate what would be the output of a single sensor on a crosswalk given a trajectory of a single pedestrian. This is how you would use the package.

Create an object SmartCrosswalk First we need to create an crosswalk with one sensor and its dimensions :

```
sensors = [(6.33, 2.33, 0)] #Microradar at (6.33, 2.33) and oriented along
    the vertical axis
dimensions = (10, 68) #10 feet wide and 68 feet long Crosswalk
```

```
SC = SmartCrosswalk(sensors, dimensions, feet=True) # Creation of the object SC
```

Listing 1: creation of a SmartCrosswalk object

If you want to simulate the outputs of sensors at the Danville intersection you can directly create the object by executing:

```
SC = createSensysCrosswalk()
```

Listing 2: creation of the Danville crosswalk object

Define the trajectory of a pedestrian We make the assumption that the pedestrian moves at a constant speed, given a list of positions, the exact trajectory of the pedestrian with respect to the time is perfectly defined. We then add this trajectory to the object. To show the set up, the user can use the object function : "show_set_up".

```
lists_of_steps = [(3, 21), (2, 10), (2, -1)] #Successions of steps of a pedestrian
SC.add_simulation_trajectory(lists_of_steps, feet=False)
SC.show_set_up()  # Show the crosswalk, sensors and trajectories
```

Listing 3: definition of a trajectory

Simulation of the output of the sensor Then once it is done, we can launch the command "run" that simulates the detection bins for each sensors according to the trajectory of the pedestrian and the parameters of the micro-radar. We can show the result of the sensors using "plot_sensors_result(name of the sensor)".

```
SC.run_simulation(v0=2, time_steps=0.2)
SC.plot_sensors_result(0)  # Plot the detection bins in function of time for sensor 0
```

Listing 4: simulation of the output of the sensor

4.4 Example 2 : Optimizing sensors arrangement

Set up pedestrian trajectories First we create a SmartCrosswalk object :

```
dimensions = (10, 68) #10 feet wide and 68 feet long Crosswalk
SC = SmartCrosswalk([], dimensions, feet=True) # Creation of the object SC
Trajectories = [(0.5, [(-0.5, 25), (3.7, -5)]), (0.5, [(3.7, 25), (-0.5, -5)])]
SC.set_up_trajectories(Trajectories)
```

Listing 5: set-up of the trajectories

Find the best k-sensors arrangement Say you want to find the best arrangement for 2 sensors. You can execute the following lines of code, that will find the arrangement, show it, save it to SC.micro_radars and finally compute the detection score.

```
SC.set_up_k_sensors(k=2, maxiter=20, show=True, save=True) #optimize arangement for 2 sensors, and show the results
logD, D = SC.compute_scores(plot=True) #compute the logD and D an show the arangement
```

Listing 6: best k-sensors arrangement

Find the best arrangement based on initial arrangement Say you want to find the best arrangement from an initial arrangement, for example of the Danville crosswalk. You can execute the following lines of code, that will find the arrangement, show it, save it to SC.micro_radars and finally compute the detection score.

```
SC = createSensysCrosswalk()
Trajectories = [(0.5, [(-0.5, 25), (3.7, -5)]), (0.5, [(3.7, 25), (-0.5, -5)])]
SC.set_up_trajectories(Trajectories)
SC.optimize_sensors(trajectories=Trajectories, initial_conditions=None, maxiter=50, show=True, save=True)
```

Listing 7: best arrangement based on initial arrangement

Results The result of this example are displayed in the appendices.

5 Conclusion

In recent years, solutions to improve road safety at intersections have emerged. One such solution is the use of micro-radars at pedestrian crossings. This solution has been tested successfully in Danville, California. The micro-radars form the basis of safety systems in this smart intersection and they focus on predicting accident risks and conflict events almost in real time. While the use of the data from these sensors has proven to be satisfactory, in this report we wanted to study the best placement of these sensors. Finding the best arrangement for a fixed number of sensors to maximise the detection rate on pedestrian trajectories is essential to have an economically efficient way to detect pedestrians and improve their safety in the context of vision Zero.

The objective of this project was to propose an optimization framework providing insight into the best way to arrange micro-radars on a pedestrian crossing. The contribution of the project is threefold. First, we proposed a model for the gain function of a micro-radar. Second, we laid the basis for an optimization program and presented some results. Finally we distributed a python package that let users simulate sensors output and optimize the micro-radars arrangement.

We modeled the radar gain function as a simple function of the distance and relative angle between the pedestrian and the sensor. From this, we were able to create a simulation tool capable of producing the detection bins of a sensor relative to a pedestrian path. For this part, some important assumptions were made to simplify the model of our micro-radar. Namely, we disregarded the effect of the Doppler shift and the sampling rate on the sensors output. Finally, we only considered a single pedestrian trajectory. We did not study the very interesting phenomenon of mixing trajectories of different users (multiple pedestrians, bicycle, car) and their effects on the sensor output. Further work should be done to validate this model.

We then argued that the integral of the radar gain function along a pedestrian path can be used as a measure of the quality of radar detection of a pedestrian. From there, we introduced a detection score and its differentiable counterpart, the logarithm of the detection score. Finally, using a quasi-Newton algorithm (BFGS), we showed that it is possible to find a better layout than the one currently used on the Danville pedestrian crossing. Of course, it should be noted that our arrangement is better in the sense of our detection score and in the sense of the set of frequent trajectories we have provided. Further work should be conducted to validate experimentally our radar gain function, and our detection score.

References

[1] National Highway Traffic Safety US Department of Transportation.

[2] Julia B Griswold, Aditya Medury, Robert J Schneider, Dave Amos, Ang Li, and Offer Grembek. A pedestrian exposure model for the california state highway system. *Transportation research record*, 2673(4):941–950, 2019.

[3] Offer Grembek, Alex A Kurzhanskiy, Aditya Medury, Pravin Varaiya, and Mengqiao Yu. Introducing an intelligent intersection. *ITS Reports, 2018 (13)*, 2018.

[4] SF Polanis. Improving intersection safety through design and operations. In *Today's Transportation Challenge: Meeting Our Customer's ExpectationsInstitute of Transportation Engineers (ITE)*, number CD-016, 2002.

[5] Jennifer Oxley, Brian Fildes, Bruce Corben, and Jim Langford. Intersection design for older drivers. *Transportation Research Part F: Traffic Psychology and Behaviour*, 9(5):335–346, 2006.

[6] Yingying Ma, Xiaoran Qin, Offer Grembek, and Zhiwei Chen. Developing a safety heatmap of uncontrolled intersections using both conflict probability and severity. *Accident Analysis & Prevention*, 113:303–316, 2018.

[7] Offer Grembek, Alex Kurzhanskiy, Aditya Medury, Pravin Varaiya, and Mengqiao Yu. Making intersections safer with i2v communication. *Transportation Research Part C: Emerging Technologies*, 102:396 – 410, 2019.

[8] Ali M Niknejad. Eecs 242: Rf mixers.

[9] Michael T Volling. A new solution to the growing problem of bicycle detection. *IMSA Journal*, 51(1), 2013.

[10] Mandy Lange, Dietlind Zühlke, Olaf Holz, and Thomas Villmann. Applications of lp-norms and their smooth approximations for gradient based learning vector quantization. In *ESANN*, 2014.

[11] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[12] Amir Beck. First-order methods in optimization. 2017.

[13] Jorge Nocedal and S Wright. *Numerical Optimization*. 2006 publisher =.

[14] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[15] Samuel J Maltby. Trisecting a rectangle. *Journal of Combinatorial Theory, Series A*, 66(1):40–52, 1994.

A Model Fitting

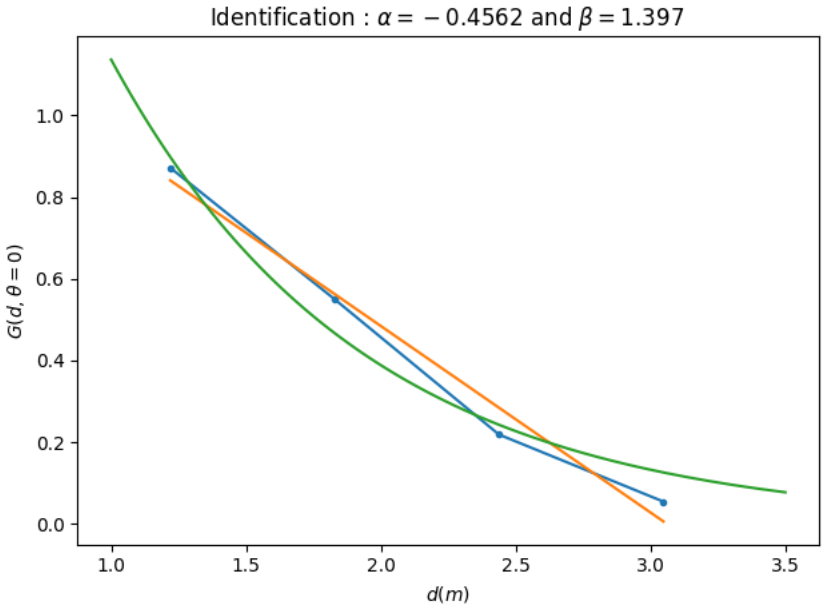


Figure 28: $A(x) = \alpha e^{\beta x}$

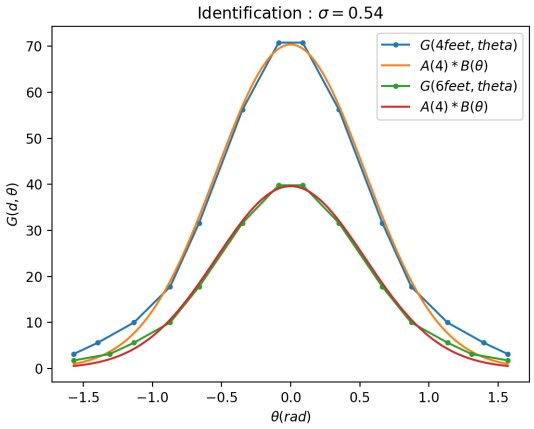
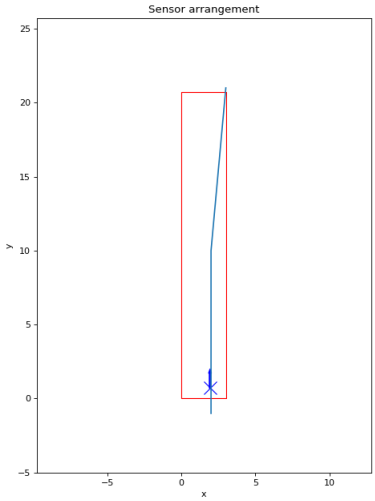
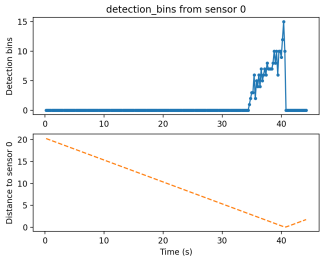


Figure 29: $B(\theta; \sigma) = e^{\frac{1}{2}\frac{\theta^2}{\sigma^2}}$

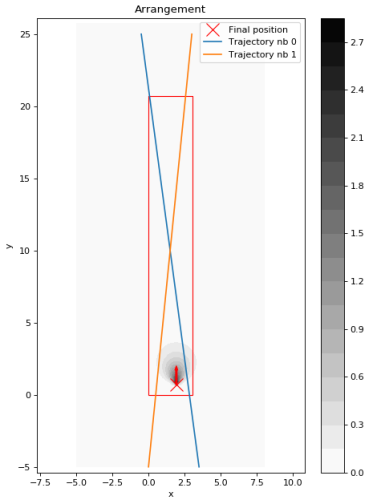
B Example 1



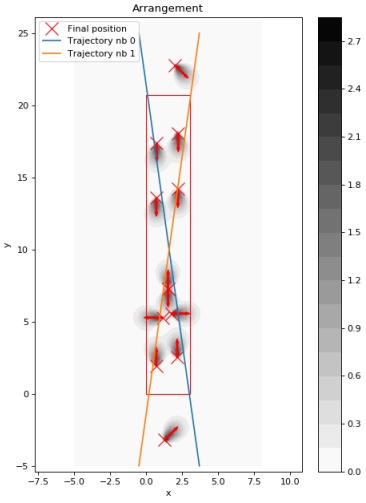
(a) Set-up of a crosswalk with one trajectory and one sensor



(b) Output of the sensor with respect to time



(a) Radar Gain Pattern for the same sensor with two trajectories



(b) Radar Gain Pattern for the Danville arangement with two trajectories

C Expression of the Gradient and Result for 1 sensor over 1 trajectory

C.0.1 Optimization of 1 sensor over 1 trajectory

To get a feeling and derive the Jacobian for multiple sensors and trajectories, let us try to figure a simple case. If we have one sensor S to place and one trajectory T, we know that we want to place our sensor on the trajectory with the angle oriented along the trajectory curve. The optimization problem is finding x_0, y_0, θ_0 to maximize :

$$\log D(x_0, y_0, \theta_0; (1, f)) = \oint_{f=0} log(g(z; x_0, y_0, \theta_0))dz$$

Along the trajectory, $f(x, y) = 0 \iff x = ay + b$, the distance from the sensor to $(x, y) \in f^{-1}(0)$ is :

$$d(y; x_0, y_0) = \sqrt{(ay + b - x_0)^2 + (y - y_0)^2} = \sqrt{P_2(y; x_0, y_0)}$$

Similarly, the angle the orientation of the sensor and $(x, y) \in \{f = 0\}$ is :

$$\theta = \text{atan2}\big(x_0 - ay - b, y_0 - y\big) - \theta_0$$

Knowing the relationship between x and y and the closed form of g, the parameters α, β, σ, we can rewrite the detection score as :

$$\log D(x_0, y_0, \theta_0; (1, f)) = l \log \alpha - \alpha \int_{y=0}^{l} d(y; x_0, y_0)\text{dy} - \frac{1}{2\sigma^2} \int_{y=0}^{l} (\text{atan2}\big(x_0 - ay - b, y_0 - y\big) - \theta_0)^2 \text{dy}$$

Let us show that $(x_0^*, y_0^*, \theta_0^*) = (a\frac{l}{2} + b, \frac{l}{2}, \arctan(a))$ is a minimizer of $\log D$. In that case, if we denote d^* the distance function from the optimum, let us notice that :

$$\frac{x - x_0^*}{y - y_0^*} = a, \quad \forall (x, y) \in f^{-1}(0) \tag{3}$$

$$\text{atan2}\big(x_0^* - x, y_0^* - y\big) = \arctan(a) = \theta^*(0), \quad \forall (x, y) \in f^{-1}(0) \tag{4}$$

$$d^*(l - y) = d^*(y), \quad \forall (x, y) \in f^{-1}(0) \tag{5}$$

We compute formally the gradient and we obtain the following result in closed form:

$$\nabla \log D(x_0, y_0, \theta_0; (1, f)) = \begin{bmatrix} \int_{y=0}^{l} -\alpha \frac{x(y) - x_0}{d(y)} - \frac{y_0 - y}{\sigma^2 d(y)^2} \left(\theta_0 - \text{atan2}\big(x_0 - ay - b, y_0 - y\big)\right) \text{dy} \\ \int_{y=0}^{l} -\alpha \frac{y - y_0}{d(y)} - \frac{x - x_0}{\sigma^2 d(y)^2} \left(\theta_0 - \text{atan2}\big(x_0 - ay - b, y_0 - y\big)\right) \text{dy} \\ \int_{y=0}^{l} -\frac{1}{\sigma^2} (\theta_0 - \text{atan2}\big(x_0 - ay - b, y_0 - y\big)) \text{dy} \end{bmatrix}$$

- First, from 4 : $\theta_0^* = \arctan(a)$, $\nabla_{\theta_0} \log D(x_0^*, y_0^*, \theta_0^*) = 0$
- Second, $\nabla_{y_0} \log D(x_0^*, y_0^*, \theta_0^*) = \int_{y=0}^{l} -\alpha \frac{y - y_0^*}{d(y)} \text{dy} = -\alpha \big[\int_{y=l/2}^{l} \frac{y - y_0^*}{d(y)} \text{dy} + \int_{y=l/2}^{l} \frac{(l-y) - y_0^*}{d(l-y)} \text{dy} \big]$, now from from 5 : $\nabla_{y_0} \log D(x_0^*, y_0^*, \theta_0^*) = 0$.
- Third, by the same mechanism, $\nabla_{x_0} \log D(x_0^*, y_0^*, \theta_0^*) = 0$

We conclude that

$$\nabla \log D(x_0^*, y_0^*, \theta_0^*; (1, f)) = 0$$

We should check that the hessian is definite positive at this point, but this part is to give a sense of the optimization and to provide formally the expression of the gradient. For a single trajectory, one optimal position is the middle of the segment with an orientation towards the trajectory. This is satisfactory since this is what we would expect for such a simple example.

C.0.2 1 sensor, p trajectories

If we have one sensor to place and p trajectories, there is no closed form solution. The optimization is now finding $S = (x_0, y_0, \theta_0)$ to maximize :

$$\log D(T_p, S) = \sum_{i=1}^{p} \omega_i \oint_{f_i=0} \log g(z; x_0, y_0, \theta_0) dz$$

Thanks to the linearity of the sum, we can quickly obtain the gradient of the function :

$$\nabla \log D(x_0, y_0, \theta_0) = \sum_{i=1}^{p} \omega_i \nabla \log D(x_0, y_0, \theta_0; (1, f_i))$$

C.0.3 k sensors, p trajectories

We already have the following identity :

$$\nabla \log D(x_0, y_0, \theta_0; (1, f)i) = \begin{bmatrix} \int_{x,y \in f^{-1}(0)} \nabla_{x_0} \log g(y; x_0, y_0, \theta_0) \mathrm{dy} \\ \int_{x,y \in f^{-1}(0)} \nabla_{y_0} \log g(y; x_0, y_0, \theta_0) \mathrm{dy} \\ \int_{x,y \in f^{-1}(0)} \nabla_{\theta_0} \log g(y; x_0, y_0, \theta_0) \mathrm{dy} \end{bmatrix}$$

In this case,

$$\log D(S; T) = \sum_{i=1}^{p} \omega_i \oint_{f_i=0} \mathcal{S}_\alpha\big(\log g(z; S))\big) dz$$

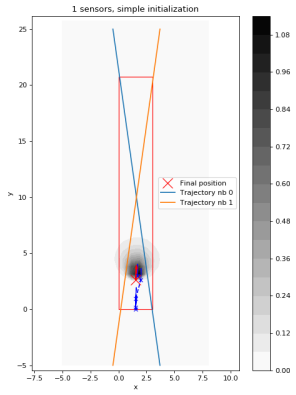
Let us note $x_{r,l}$ the $l \in \{1, 2, 3\}$ parameter of the rth sensor. According to the chain rule :

$$\begin{aligned} \frac{\partial}{\partial x_{r,l}} \mathcal{S}_\alpha\big(\log g(z; S))\big) &= \frac{\partial \log g(z; x_{r,\cdot})}{\partial x_{r,l}} \frac{\partial}{\partial \log g(z; x_{r,\cdot})} \mathcal{S}_\alpha\big(\log g(z; S))\big) \\ &= \frac{\partial \log g(z; x_{r,\cdot})}{\partial x_{r,l}} \frac{\mathrm{e}^{\alpha \log g(z; x_{r,\cdot})}}{\sum_{i=1}^{p} \log g(z; x_{i,\cdot})} \big[1 + \alpha[\log g(z; x_{r,\cdot}) - \mathcal{S}_\alpha\big(\log g(z; S))\big)]\big] \end{aligned}$$

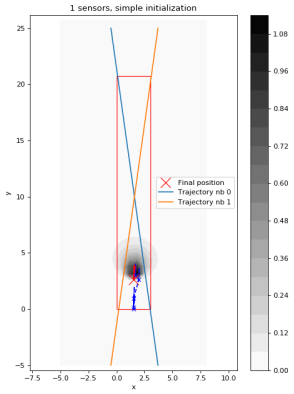
Therefore, we have the gradient with respect to every parameter :

$$\nabla_{x_{r,l}} \log D(S; T) = \sum_{i=1}^{p} \omega_i \oint_{f_i=0} \nabla_{x_l} \log g(z; x_{r,\cdot}) \frac{\mathrm{e}^{\alpha \log g(z; x_{r,\cdot})}}{\sum_{i=1}^{p} \log g(z; x_{i,\cdot})} \big[1 + \alpha[\log g(z; x_{r,\cdot}) - \mathcal{S}_\alpha\big(\log g(z; S))\big)]\mathrm{dz}$$

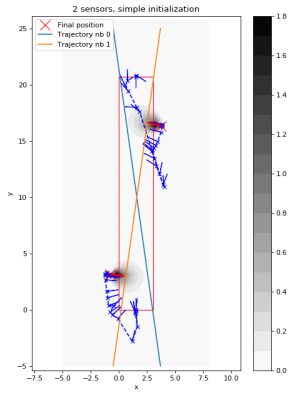
D Best arrangements



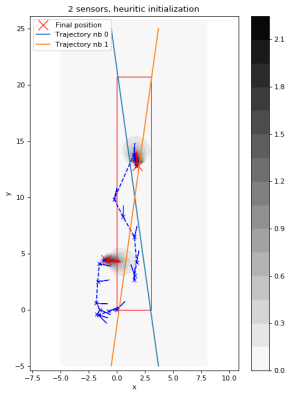
(a) Final result for 1 sensor and simple initialization



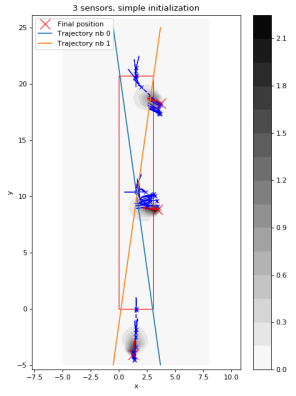
(b) Final result for 1 sensor and simple/heuristic initialization



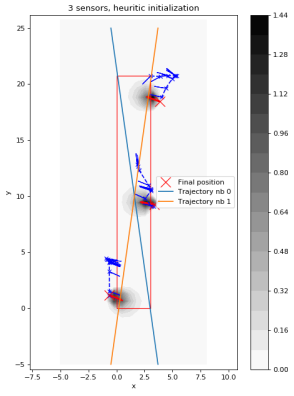
(c) Final result for 2 sensors and simple initialization



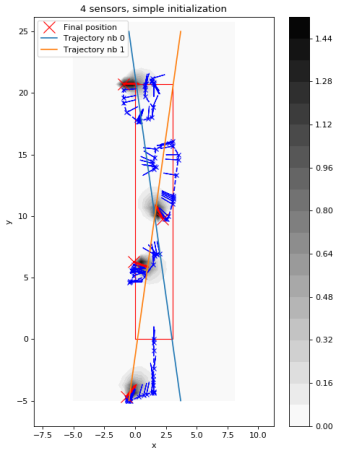
(d) Final result for 2 sensors and heuristic initialization



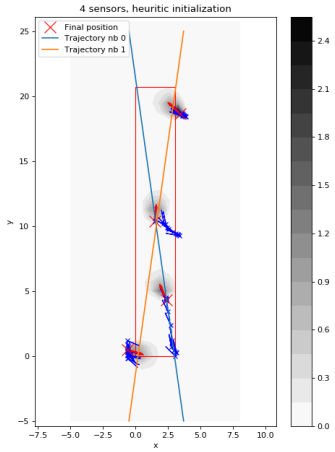
(e) Final result for 3 sensors and simple initialization



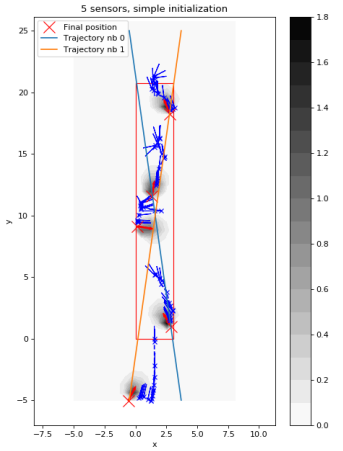
(f) Final result for 3 sensors and heuristic initialization



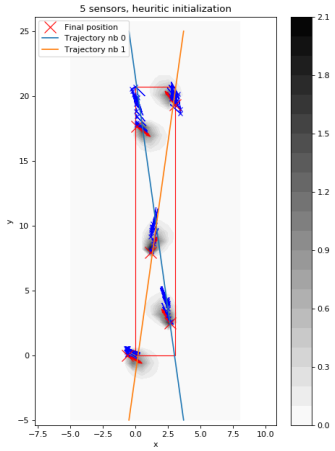
(a) Final result for 4 sensors and simple initialization



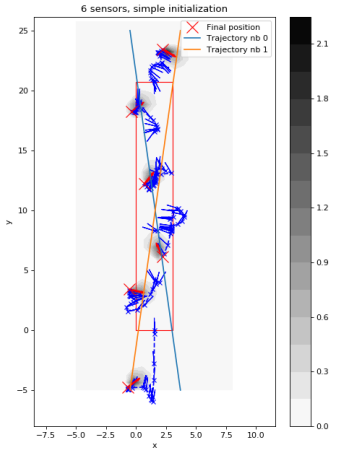
(b) Final result for 4 sensors and heuristic initialization



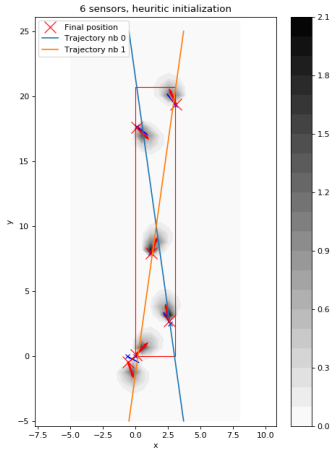
(c) Final result for 5 sensors and simple initialization



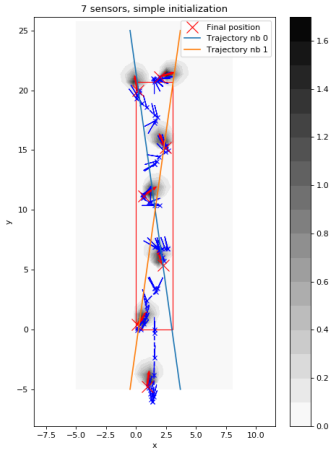
(d) Final result for 5 sensors and heuristic initialization



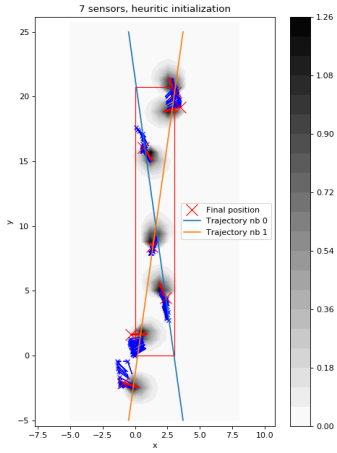
(e) Final result for 6 sensors and simple initialization



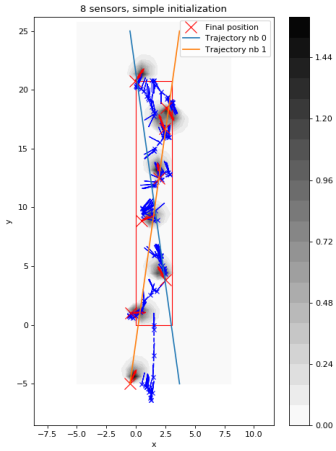
(f) Final result for 6 sensors and heuristic initialization



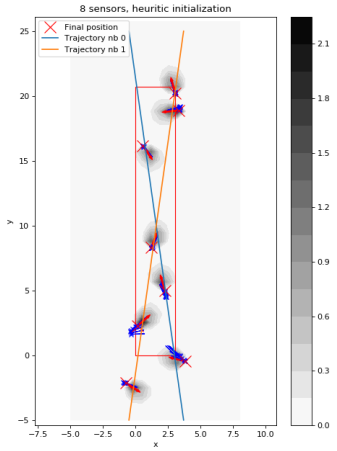
(a) Final result for 7 sensors and simple initialization



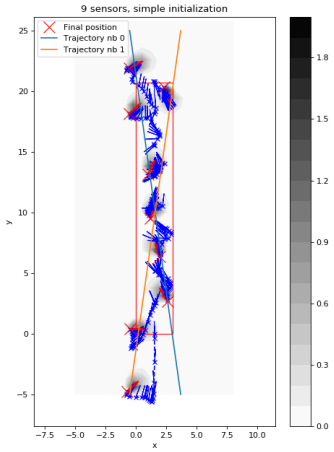
(b) Final result for 7 sensors and heuristic initialization



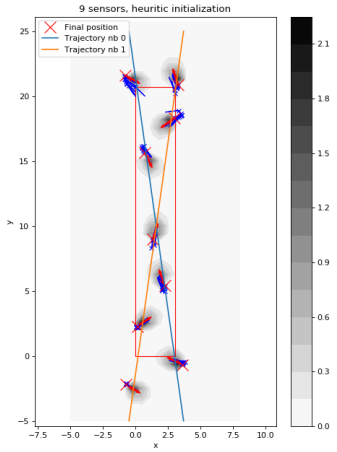
(c) Final result for 8 sensors and simple initialization



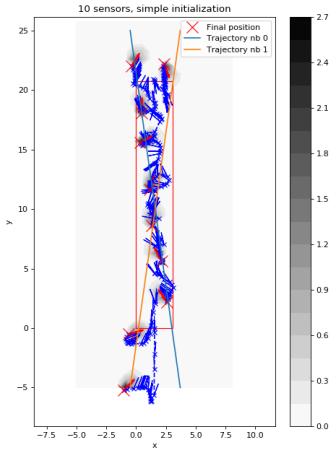
(d) Final result for 8 sensors and heuristic initialization



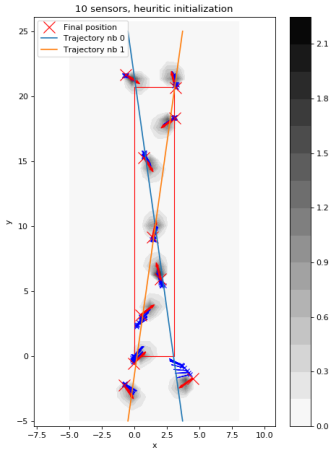
(e) Final result for 9 sensors and simple initialization



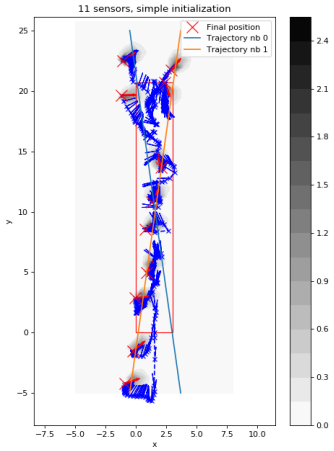
(f) Final result for 9 sensors and heuristic initialization



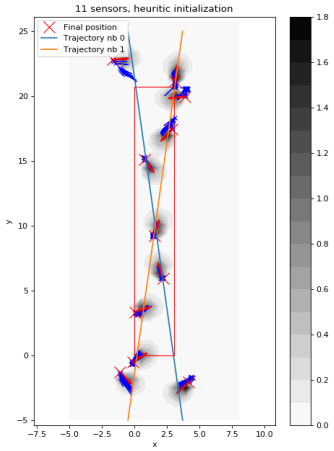
(a) Final result for 10 sensors and simple initialization



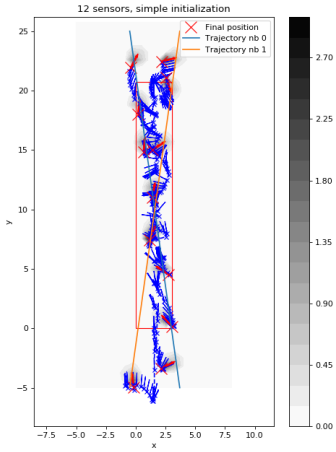
(b) Final result for 10 sensors and heuristic initialization



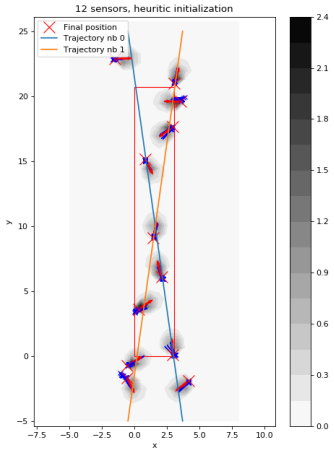
(c) Final result for 11 sensors and simple initialization



(d) Final result for 11 sensors and heuristic initialization



(e) Final result for 12 sensors and simple initialization



(f) Final result for 12 sensors and heuristic initialization