STATUS REPORT

Surveillance Of The Bay Of Biscay

Written by :

Elouan Autret	Thomas Boulier
Maxime Bouyssou	Philippe Chuzel
Alice Danckaers	David Duverger
Raphael Finkelstein	Sylvain Hunault
Pierre Jacquot	Mael Le Gallic
Eric Mourre	Thiago Oliveira
Benoit Raymond	Khadimoullah Vencatasamy

Under the supervision of :

Luc Jaulin
Benoit Zerr

ENSTA Bretagne

Option Systèmes Perception Information Décision

Contents

Acknowledgment

Chapitre 1

Introduction

The goal of this project is to ensure the security of the Bay of Biscay by detecting any intruder entering this maritime area. A group of robots, equipped with GPS, are being used thanks to specific algorythms and a particular monitoring strategy.

In order to proceed with the project, the following working hypotheses are considered :

- The simulation can be done in a two-dimensional world ;
- Intruders will have a constant given velocity throughout the simulation ;
- Every surveillance robot has a detection range of d_i within which an intruder is necessarily detected ;
- The safe zones are regarded as a group of rectangles where the intruder can not be.

Two deliverables will be delivered :

- The first one is a theoretical simulation written to a large extent in Python 3.0 which will allow to determine whether or not the used algorythms are relevant ;
- The second one is a practical simulation using some buggies robots available in the robotics group of ENSTA Bretagne.

Chapitre 2

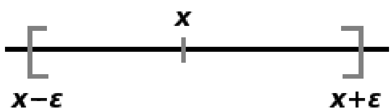
Secure zone processing

2.1 Introduction to interval analysis

David DUVERGER

For this project, we have chosen the intervals calculation approach which guarantees solutions in a given interval of uncertainty.

In the Intervals theory, a known value is replaced by an interval with a certain uncertainty in which it is sure that this measure is include. So, for a known value x, the corresponding interval is $[x - \epsilon, x + \epsilon]$.



The goal of this method is to give the percentage of certainty which allows to obtain reliable and robust results. In tow dimensions, intervals are represented by boxes. Here is an example of the resolution by Intervals. It is the result of S1|S2 with :

$$
\begin{aligned}
\mathbb{S}_1 &= \left\{(x,y) \in \mathbb{R}^2 \mid \ (x-1)^2 + (y-2)^2 \in [4,5]^2\right\} \\
\mathbb{S}_2 &= \left\{(x,y) \in \mathbb{R}^2 \mid \ (x-2)^2 + (y-5)^2 \in [5,6]^2\right\}.
\end{aligned}
$$

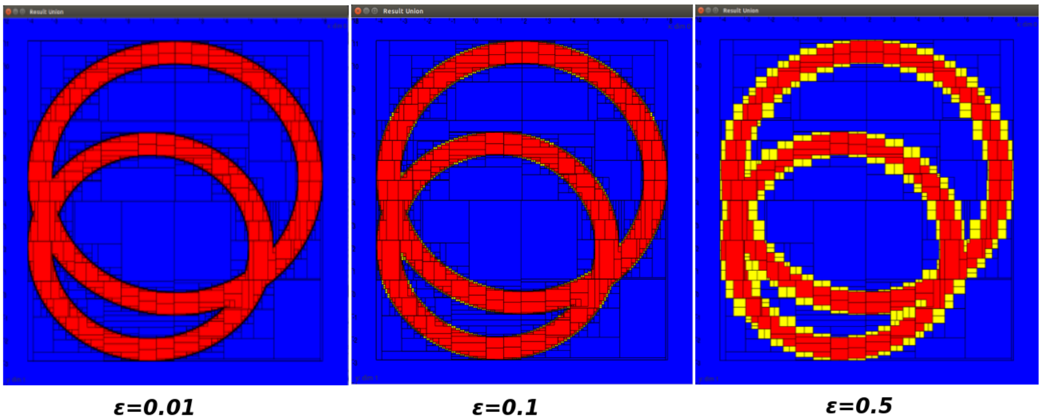


FIGURE 2.1 – Resolution of S1 U S2 for different precision with intervals calculation

If a box frames a solution, it is cut in two parts and the calculation is reiterated on these two boxes which allow to discriminate one of them and to reduce solutions. However, this operation take lot of time because the number of boxes increase of 2n each time. That is why, the contractors are used, they allow to refine the boxes at the intervals which include strictly the solution before to cut them. This method allow to have an important gain of time during the simulation.

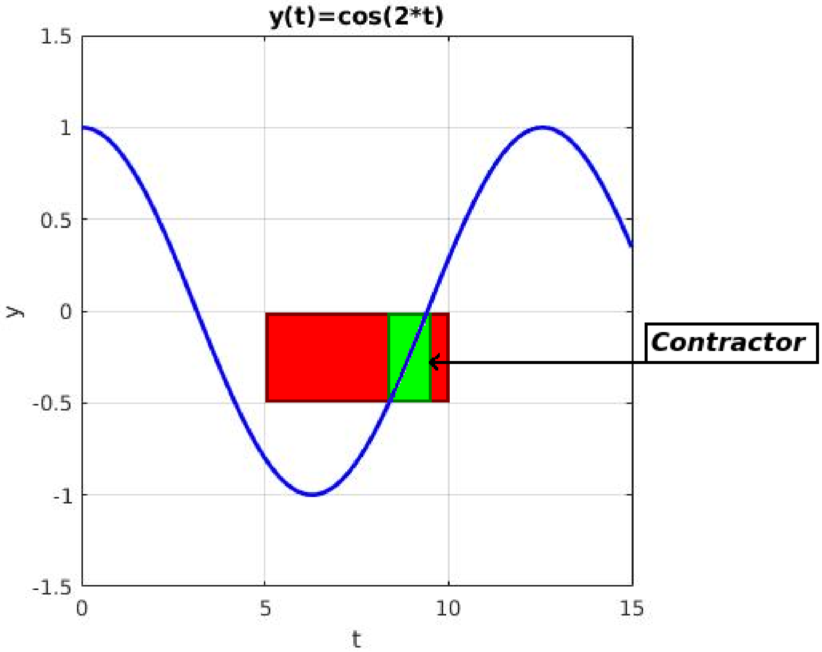


FIGURE 2.2 – Example of the utilisation of contractor

For the example, the signal cos(2*t) in blue is used. The box in red corresponds to the normal box and the green box corresponds to the contractor. With this example, we can easily see that contractors improve the simulation in terms of time.

For the simulation of the Gascogne project, we used a SIVIA function which make the paving for the representation of the France, robots control and their erosion.

2.2 Secure zone estimation with interval analysis

Elouan AUTRET

2.2.1 Thick Functions

A thick function is a function of $\mathbb{R}^n$ in $\mathbb{R}^p$ that associate to an element x $\in \mathbb{R}^n$ a convex element of $\mathbb{R}^p$ where there is intervals in the parameter of the function itself

For example considering the function f :
f : $\mathbb{R}^2 \rightarrow \mathbb{R}^2$
$\mathbf{x} \rightarrow \mathbf{x} + [\text{-}1, 1]$ =0
Even if the width of x is null (ex : [1,1]) the width of f(x) will be strictly greater than zero ([0,2]). Those thick functions allow the modelling of the uncertainty of the position of the boat which will watch the bay.
Indeed if [1,2] is the position of the boat and 10 is the range of detection by the boat, then the set of the secure zone would be originally :

$$\{x \in \mathbf{R}^2 | (x(1) - 1)^2 + (x(2) - 2)^2 \leq 100\}$$

But if we add an uncertainty of 0.5 to the position then the set become :

$$\{x \in \mathbf{R}^2 | (x(1) - [0.5, 1.5])^2 + (x(2) - [1.5, 2.5])^2 \leq 100\}$$

This set can easily be found with interval analysis using the ibex library with a separator for the precedent equation then proceeding with the SILVIA algorithm and by associating the code VIBES we can visualize the zones reached by the boat :

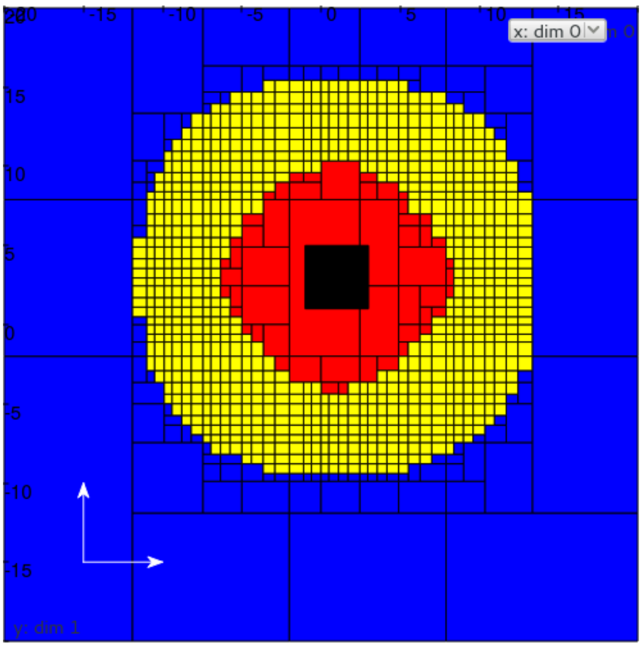


FIGURE 2.3 – Zone secured by one boat in red, yellow zone is uncertain,blue is not secured.

The figure 2.3 shows the zone covered by the surveillance boat, in red the zone is certain to be covered, in yellow it is not sure it depends on the real place of the boat in the black square or uncertainty.

2.2.2 Test for Biscay Bay Surveillance

For all this section we consider that $\mathbf{X} \in \mathbb{R}^2$ is included in the bay of Biscay (the set $\mathbb{G}$) to facilitate the comprehension.
The figure 2.3 shows that the zone covered by the boats can be found but it also point out that the computing of the zone is not efficient, the uncertain zone is considered a border to the secured zone so the SIVIA algorithm cut the boxes to their minimal even if we know their status thus proceeding to more computation than needed, a new test is required instead of a simple separator to permit a faster and cleaner computation of the secured zone. The test can be found in the algorithm 1 which take a box (an element of $\mathbb{R}^2$)) and determine its status, the state can vary between six status where for the moment the border regroup more or less three status as it can be seen in the following table :

Symbol	In Algorithm	Meaning
0	OUT	Box is not in the Secure Zone
1	IN	Box is in the secure zone
?	UNKNOWN	Box is at the border of the secure zone
[0, ?]	UNKNOWN	Box is on the external border of the covert zone
[?,1]	UNKNOWN	Box is on the internal border of the covert zone
[0,1]	MAYBE	Box is in the uncertainty zone

Algorithm 1 Is $\mathbf{X} \subseteq \mathbb{S}_m$, $\mathbb{S}_m$ = Secured Zone by boat m and $\mathbf{X} \in \mathbb{R}^2$

Require: X,$range$ (reach of boat),m (position of boat)

$Xmx \leftarrow max([0,0], sign((X[0]-m[0].ub()) \cdot (X[0]-m[0].lb()))) \cdot min((X[0]-m[0].lb())^2, (X[0]-m[0].ub())^2)$

$Xmy \leftarrow max([0,0], sign((X[1]-m[1].ub()) \cdot (X[1]-m[1].lb()))) \cdot min((X[1]-m[1].lb())^2, (X[1]-m[1].ub())^2)$

$Xm \leftarrow Xmx + Xmy$

$Xp \leftarrow max((X[0]-m[0].lb())^2, (X[0]-m[0].ub())^2) + max((X[1]-m[1].lb())^2, (X[1]-m[1].ub())^2)$

$Xub \leftarrow Xp|Xm$

if $Xub \cap [0, \text{range}^2] = \emptyset$ **then**

  return OUT

else if $Xub \subseteq [0, \text{range}^2]$ **then**

  return IN

else

  if $\text{range}^2 - Xp.ub() < 0]$ **then**

    if $Xm - \text{range}^2 \subseteq [-\infty, 0]$ **then**

      return MAYBE

    else

      return UNKNOWN (later :UNKNOWN2)

    end if

  else

    return UNKNOWN

  end if

end if=0

The solution of the zone covered by the boat resemble to the figure 2.4, where by seeing the size of the boxes in the uncertain zone it is sure that the computation was efficient :

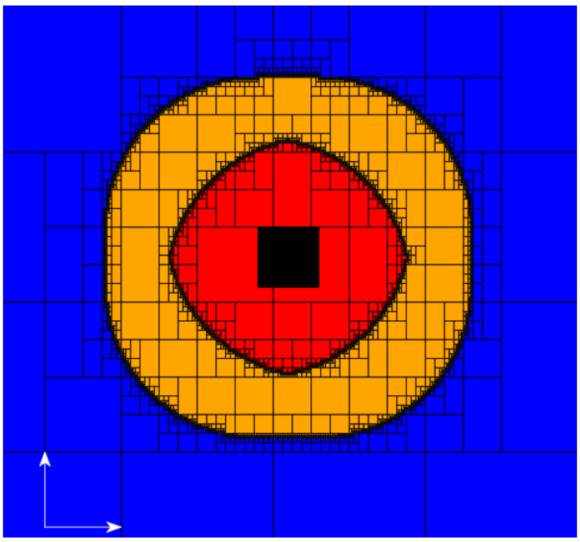


FIGURE 2.4 – Zone secured by one boat in red, orange zone is uncertain, yellow is the border and blue is not secured.

This table was temporary has it was omitting the overlapping of the different boat as seen in the figure 2.5, the external border of the zone covered by on boat is overlapping the uncertain zone of a different boat :

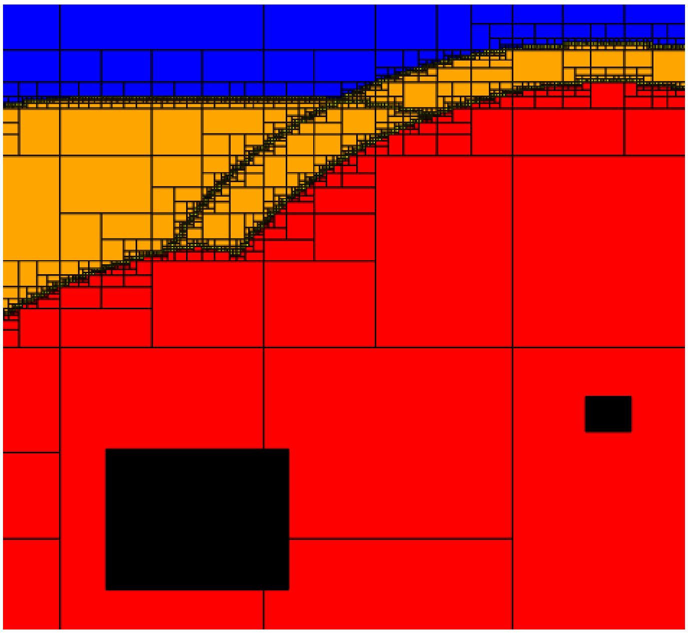


FIGURE 2.5 – Zone secured by two boat in red, orange zone is uncertain, yellow is the border.

In order to avoid those wrong and useless computations, the algorithm 1 need to take into account the fact that an external border of a zone can be in an uncertain zone this change the correspondence table to :

Symbol	In Algorithm	Meaning
0	OUT	Box is not in the Secure Zone
1	IN	Box is in the secure zone
?	UNKNOWN	Box is at the border of the secure zone
[0, ?]	UNKNOWN2	Box is on the external border of the covert zone
[?,1]	UNKNOWN	Box is on the internal border of the covert zone
[0,1]	MAYBE	Box is in the uncertainty zone

Now the algorithm 1 need to update this is done by changing the return from UNKNOWN to UNKNOWN2 in operand 15.
But the result seen in figure 2.5 need to fusion result of algorithm 1 between the different boat, this can be done by evaluating the result two by two boats, wich correspond to the Truth table that follow :

Test1/Test2	0	1	?	[0, ?]	[?,1]	[0,1]
0	0	1	?	[0, ?]	[?,1]	[0,1]
1		1	1	1	1	1
?			?	?	[?,1]	[?,1]
[0, ?]				[0, ?]	[?,1]	[0,1]
[?,1]					[?,1]	[?,1]
[0,1]						[0,1]

This truth table is translated in an algorithm(see 2).

Algorithm 2 Is $\mathbf{X} \subseteq \mathbb{S}$, $\mathbb{S}$ = Secure Zone and $\mathbf{X} \in \mathbb{R}^2$

Require: X,$range$ (reach of a boat),M (list of position of boats)

$R \leftarrow$ {Algorithme 1(X,range,m)}$_{m \in M}$
if IN $\subset R$ **then**
  return IN
else if UNKNOWN $\subset R$ **then**
  return UNKNOWN
else if MAYBE $\subset R$ **then**
  return MAYBE
else if $\forall r \in R$, r = OUT **then**
  return OUT
else
  return UNKNOWN
end if=0

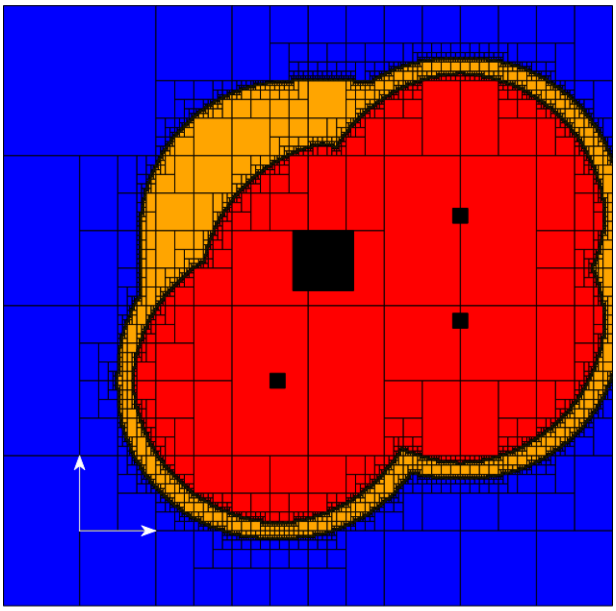


FIGURE 2.6 – Zone secured by four boat in red, orange zone is uncertain,blue is not secured.

Those last change make the computation of the zone more efficient and correct, in figure 2.6 there is no more overlapping of the external border.

If the uncertain zone is not needed the computation of the algorithm SIVIA can be accelerated by ignoring it. This can be done by changing the return value in the operand 15 in the algorithm 1 from UNKNOWN2 to OUT and the return value in the operand 13 from MAYBE to OUT (the second change will not improve efficiency but just place the box in the right set), this gives the following result :

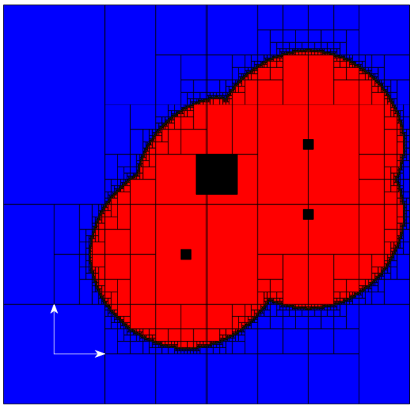


FIGURE 2.7 – Zone secured by four boat computed with algorithm 2 and 1.

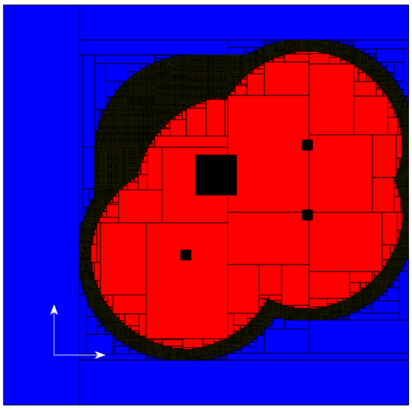


FIGURE 2.8 – Zone secured by four boat computed with separators

The result in figure 2.7 can also be obtain with a the combination of separators (see figure 2.8 but the result will be longer to get as it would cut through the uncertain zone even if it is computation time wasted. To demonstrate the waste of time done by the separators the figure 2.8 has taken thirty six times the time taken to compute the figure 2.7 (fastest computation times).

2.3 Mesh to boxes functions

2.4 Erosion functions

2.5 Global Sivia algorithm

2.6 SIVIA Algorithm

David DUVERGER

I will present here the fusion of the paving of the Gascogne Golf, the robots and their erosion. So, it is the resolution of the following mathematical formula :

$$X(t) = G \cap F_\delta(X(t-\delta)) \cap g^{-1}([di, \infty])$$

Algorithm 3 SIVIA algorythm

Inputs

0: Area $X0$
0: Object $France$
0: Object $Erosion$
0: Object $Robots$ =0

Output

0: Boxes $FranceURobotsUErosion$
0: Boxes $France$
0: Boxes $RobotsUErosion$
0: Boxes Sea =0

Initialization

0: $stack \leftarrow deque([IntervalVector(X0)])$
0: $lf \leftarrow LargestFirst(eps/2.0)$
0: $k \leftarrow 0$
0: $BoxesFranceURobotsUErosion \leftarrow []$
0: $BoxesFrance \leftarrow []$
0: $BoxesRobotsUErosion \leftarrow []$
0: $BoxesSea \leftarrow []$ =0

Algorithm 4 SIVIA algorythm (continued)

while len(stack) **do**

$X \leftarrow stack.popleft()$
$FranceTest \leftarrow France.function(X)$
$ErosionTest \leftarrow Erosion.function(X)$
$RobotsTest \leftarrow Robots.function(X)$

//Otain the 4 corners of each box in pixel :
$i, j \leftarrow France.toPixels(X[0].lb(), X[1].ub())$
$i1, j1 \leftarrow France.toPixels(X[0].ub(), X[1].ub())$
$i2, j2 \leftarrow France.toPixels(X[0].ub(), X[1].lb())$
$i3, j3 \leftarrow France.toPixels(X[0].lb(), X[1].lb())$

//If we are in the France, robots or erosion :
if FranceTest==IBOOL.IN or ErosionTest == IBOOL.IN or RobotsTest == IBOOL.IN **then**
$BoxesFranceURobotsUErosion.append([[i, j], [i1, j1], [i2, j2], [i3, j3]]);$

//If we are in the France :
if FranceTest==IBOOL.IN **then**
$BoxesFrance.append([[i, j], [i1, j1], [i2, j2], [i3, j3]]);$
end if

//If we are in the robots or erosion :
if RobotsTest == IBOOL.IN or ErosionTest == IBOOL.IN **then**
$BoxesRobotsUErosion.append([[i, j], [i1, j1], [i2, j2], [i3, j3]]);$
end if

//Else if we are outside all :
else if I **then**f FranceTest== IBOOL.OUT and ErosionTest== IBOOL.OUT and RobotsTest == IBOOL.OUT
$BoxesSea.append([[i, j], [i1, j1], [i2, j2], [i3, j3]]);$
else
try :
if $X.maxdiam() \geq eps$ **then**
$(X1, X2) \leftarrow lf.bisect(X)$
$stack.append(X1)$
$stack.append(X2)$
end if
except Exception :
$print(type(lf), lf)$
end if
end while

$vibes.axisEqual()$
return BoxesFranceURobotsUErosion,BoxesFrance,BoxesRobotsUErosion,BoxesSea =0

With the help of this algorythm we can have a paving of the intersection between the Gascogne Golf, the robots and their erosion. We arrive to obtain the following result :

SimulationFusion.png

FIGURE 2.9 – Paving of the intersection between the Gascogne Golf, the robots and their erosion