

Hybrid Algorithm combining Genetic algorithms and CSP approach to plane an Optimized path for a Mobile Robot moving under Time Window

¹Hayet Tlijani, ²Tlijani Hatem, ³Knani Jilani, ⁴M'sirdi Nacer Kouider

(LARA, National School of Engineering of Tunis, Tunisia)

(LSIS, Ecole Polytechnique Universitaire de Marseille, France)

Abstract: The problems of optimization are considered as a type of a constraint satisfaction problem: COP (Constraint Optimization problem). In this frame goes our first works which are interested in the optimization of a reference trajectory of a mobile robot used for the surveillance (Surveyor Robot). Indeed, in the previous work we have focused on the minimization of the length of this trajectory by using the approach of the genetic algorithms. At the level of our present researches we shall add temporal windows to limit the time delayed in the passage between one surveyed site and the following one. The consideration of temporal aspects requires a modelling of problem with a satisfaction of constraints. So that, a hybridization of the algorithm of optimization already developed at the previous work by the approach of CSP (Constraint Satisfaction Problem), is indispensable for our case. The new algorithm which will be developed and feigned at this work is a genetic algorithm allowing minimizing the length of a trajectory crossed by a mobile robot of surveillance by respecting the time fixed to take during the passage through two successive sites.

Keywords-Constraint Satisfaction Problem, Genetic algorithms, Meta-heuristics, Mobile Robot, Neighbourhood method, Optimization, Path Tracking.

I. Introduction

Hybrid systems, involving intelligent paradigms, get increasing interests because of their significance in theory and application. In this paper, we present a hybrid genetic algorithm by CSP techniques to solve the optimisation path tracking problem of mobile robots that moves under time constraint.

Indeed, in the previous work [1] we have focused on the minimization of the length of this trajectory by using the approach of the genetic algorithms. At the level of our present researches we shall add temporal windows to limit the time delayed in the passage between one surveyed site and the following one. The consideration of temporal aspects requires a modelling of problem with a satisfaction of constraints. So that, a hybridization of the algorithm of optimization already developed at the previous work by the approach of CSP (Constraint Satisfaction Problem), is indispensable for our case.

The new algorithm is a hybrid algorithm allowing minimizing the length of a trajectory crossed by a mobile robot of surveillance. Furthermore it respects the time fixed to take during the passage through two successive sites. Some operators of the genetic algorithm used in the optimisation are changed. For example the crossover was changed to verify the temporal constraint. However the operator of mutation was conserved as it was used in the previous works.

The present paper is organized as follows: Section II describes the constraint satisfaction problem related to our case. Therefore it illustrates the different variables needed for the mathematic formulation of this problem. Section III gives the motion of the used mobile robot and its mathematic model. Section IV defines the heuristics and the generated algorithm to solve the mentioned problem.

II. Problem description

Alternately [2], a heuristics of selection in case of time constraint, can be reckoned by building a graph of navigation corresponding to the projection of states on the facts relative to the position of the vehicle. The costs of the arcs of this graph in terms of time or length are calculated by a planner of road: a graph of planning is built and used to estimate the consumption of time or resource necessary for the reaching of every objective.

In the same idea we can develop the case of the movement of a mobile robot in its workspace. Then we can use a planner of road that estimates a reference trajectory for the studied mobile robot. The cost estimated for this trajectory corresponds to the crossed distance. A heuristics of optimization is used to have a minimal length for this trajectory. It was the genetic algorithms approach that we have engendered in a previous work [1]

The graph studied by the planner for the generated reference path connects (binds) two objectives to achieve by the mobile robot by an arc. The objectives correspond in reality to the various artificial beacons marking the map of the workspace of our mobile robot. The passage of an objective (position) in the other one is restricted with a Time Windows.

Consequently, our problem converges in a definition of a Constraint Satisfaction Problem (CSP). Further than the minimization of the reference trajectory, we have to use the techniques of the CSP. So that we combine two approach: the genetic algorithms and the CSP techniques to resolve our problematic that minimize the distance crossed along a reference trajectory of a mobile robot constrained with a time windows.

The presented work deals with the hybridization between the genetic algorithms and techniques of Constraint Satisfaction Problem (CSP) to have a solver to the mentioned problematic.

II.1. Variables of the problem

Let N a set of k knots corresponding to the artificial beacons' positions in the configuration space of the mobile robot. These beacons mark the sites to survey by the studied mobile robot. Two successive knots n_k and n_{k+1} represent an arc of the graph that defines the movement of the mobile robot from an objective to the other one, see Fig.1.

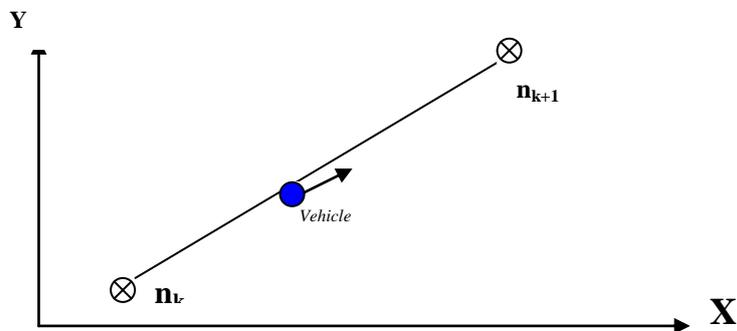


Figure 1: Knots Succession

Thus at a fixed moment the state is defined by:

- The position of artificial beacon to which the mobile robot is going;
- The date t ;
- The posture of the mobile robot (position and speed): described in the section 3 of this paper;
- The duration T between two successive positions: it determines the time windows.

II.2. Relation between two successive states

The equation (5) illustrates the relation between two successive states of the mobile robot. This relation describes the posture of the mobile robot (vehicle) in the knot n_{k+1} : its position in the configuration space, its linear velocity and angular velocity of the mobile robot. This relation defines the localization of mobile robot and how to reach the next position in function of time. This data is the variable according to which we shall choose the next objective.

II.3. Formalization of the problem

A very important part of solving problems using constraints is modelling the problem in terms of constraints, which means transforming the problem description from the natural language to the language of constraints. Both choosing the right model and choosing the right constraint satisfaction algorithm is crucial for efficient solving of the problem.

We propose in the following sections a formalism allowing describing completely the problem. This formalism generates a solver for the problem of optimization of the mobile robot path tracking with time window time.

II.3.1 Mathematical formulation

We must define the various necessary variables to make the mathematical formulation of the problem of the mobile robot's navigation in its configuration space.

- d_{ij} = Distance between the artificial beacons,
- n = Number of the artificial beacons in the space of work,

x_{ij} = binary Variable $\begin{cases} = 1 & \text{if the beacon (surveyed site) } j \text{ is visited immediately after the beacon } i ; \\ = 0 & \text{else.} \end{cases}$

The problem consists in minimizing the length of the path tracking crossed. The objective function is:

$$\text{Minimize: } D = \sum_{i=1}^n \sum_{j=1}^n d_{ij} x_{ij} \quad (1)$$

Constraints:

$$\sum_{j=1}^n x_{ij} = 1 \quad (2)$$

$$\sum_{i=1}^n x_{ij} = 1 \quad (3)$$

$$t_{n_{k+1}} - t_{n_k} \leq T \quad (4) \quad k=1, \dots, n.$$

The equation (2) defines the constraint which makes sure that we go out only once of each beacons (surveyed sites). Whereas the equation (3) describes the constraint which verifies that we enter only once every surveyed site. These constraints allow visiting all the sites only once. The temporal constraint is illustrated by the equation (4) where T the period of the time window; and t_{n_k} and $t_{n_{k+1}}$ the moments of passage of the mobile robot respectively by the position k and k+1.

According to (4) we can classify the problem held in this work as a variant of the problems of tours of vehicles: Vehicle Routing Problem with Time Windows (VRPTW). By definition T form a time window which limit the passage between two sites marked by artificial beacons in the working space of the mobile robot. The constraint of the time windows indicates that any sites cannot be visited in any order in the path. It allows the mobile robot to plan better the site following from a current position.

II.3.2 CSP definition

[3][4] Constraint satisfaction problems (CSPs) are mathematical problems where one must find states or objects that satisfy a number of constraints or criteria. It's defined in [5] as following:

A constraint satisfaction problem is defined by a triple (X, D, C_i) , where X is a set of variables, D is a domain of values, and C is a set of constraints $C_1(S_1) \dots C_n(S_n)$ where each S_i is a set of variables. A constraint C_i is a combination of valid values for the variables S_i . A solution to the CSP is an assignment of values to $S_1 \dots S_n$ that satisfies all constraints.

In our case the triple is defined as following:

$X = \{(x_{bi}, y_{bi}) / \text{position of a surveyed site marked by an artificial beacon ; } i=1, \dots, n\}$,

$D = \{b_i / b_i \in \mathbb{N} : \text{number of each beacon in the path}\}$,

$C = \{\text{equation (2), equation (3), equation (4)}\}$.

The equations ((2), (3) and (4)) form the constraints imposed on the movement of the mobile robot in its configuration space. These equations were later developed in section 2.3.1.

III. Localization and the posture of the mobile robot

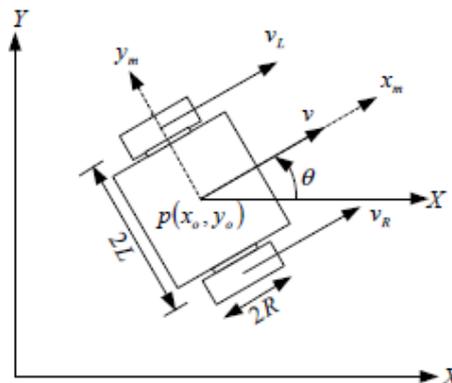


Figure 2: Posture definition of a two _wheeled mobile robot

As mentioned in the previous work [6], a two-wheeled mobile robot is considered and its structure is described by the figure Fig.2 [7], where X-Y is the global coordinates and xm-ym is the local coordinates which is fixed to the robot with its center p as the origin. As shown in Figure 2, its body is of symmetric shape and the center of mass is at the geometric center p of the body. R is the radius of the wheel and L is the displacement from the center of robot to the center of wheel. The set (x_o, y_o) represents the position of the geometric center p in the world X-Y coordinates, and the angle θ indicates the orientation of the robot. The angle θ is taken counter clockwise from the X-axis to the xm-axis.

The two wheels are fixed and controlled independently by two motors. For this reason, the considered wheeled robot is equipped with motors that are driven by on board computer. It is assumed that the wheeled mobile robot is made of a rigid frame, non-deformable wheels that do not slip, and it is moving on a horizontal plane.

In the rest of our work and realized test, we assume this mobile robot to a one particle in its configuration space. This later is equipped by a number of artificial beacons (land marks) that mark the different points to visit.

According to this description, and the mobile robot localization using artificial landmarks studied in [7] [8] we can define the posture of the used mobile robot by the following equation (5):

$$X_{k+1} = \begin{Bmatrix} x_k + v_k \cos \theta dt \\ y_k + v_k \sin \theta dt \\ v_{k+1} \\ \theta_k + \dot{\theta}_k dt \\ \dot{\theta}_{k+1} \end{Bmatrix} \quad (5)$$

Where:

X_{k+1} : the vector of the posture at the state k+1 generated in function of the previous state;

(x_k, y_k) : the position of the center of the mobile robot at the state k;

v_{k+1} : The linear velocity of the mobile robot at the state k+1;

θ_k : The angular position of the center of the mobile robot at the state k;

$\dot{\theta}_{k+1}$: The angular velocity of the mobile robot at the state k+1;

This vector describes the posture of the mobile robot in its configuration space referring to its previous one. So that we can specify at any time the different coordinates of the studied mobile robot. In dead, we will use this vector after in the algorithm 4 to calculate the position that respects the temporal constraint defined in the section II.3.2.

In this case we notice that the linear velocity is constant but we have changed by the following equation:

Linear velocity equation (6):

$$v_{k+1} = \frac{d_{k+1,k}}{\Delta t} \quad (6)$$

$d_{k+1,k}$: Distance between the positions of the mobile robot at the state k+1 and k;

And the same thing for the angular velocity:

Angular velocity equation (7):

$$\dot{\theta}_{k+1} = \frac{\theta_{k+1,k}}{\Delta t} \quad (7)$$

$\theta_{k+1,k}$: The angular between the positions of the mobile robot at the state k+1 and k;

IV. Heuristics and used algorithms

IV.1. The basic algorithm: overview

The first idea is to look for a first eligible optimal solution by applying an evolutionary algorithm (genetic algorithms).

Algorithm 1

```

randomInit population ;
For (i = 1 to n) do
    Select (individual1)
    
```

```
Select (individual2)
Individual-child= Crossover (individual1, individual1)
If (fitness (individual-child)>=fitness (individual2) then
    Replace (individual2, individual-child)
End if
If (fitness (individual-child)>fitness (individual1) then
    Replace (individual1, individual-child)
    Mutate (individual1)
End if
SortPopulation()
End for
```

The basic algorithm (Algorithm 1) is an application of the genetic algorithms. It was studied; developed and tested at the research work in [1]. The proposed algorithm applies a recombination of two better parents to every generation. If the obtained individual is better than the best parent it will be inserted. A mutation is applied to the worst parent if the resultant individual is worse than both parents. After n iteration of this process we shall have the optimal path of reference according to the objective function already defined in the paragraph II.3.1.

IV.2 The genetic algorithms operators

Genetic algorithm is particularly easy to implement and promises substantial gains in performance. Its basic idea is to keep several sub-populations that are processed by genetic algorithm. Furthermore, a migration mechanism produces a chromosome exchange between sub-population.

The main characteristics of the proposed GA consist of the following:

- A string of number encoding a feasible path joining all detected beacons in the work-space;
- A gene in the chromosome is an artificial beacon in the configuration space of the mobile robot;
- Each path is an individual that has a length;
- The length of the path is a fitness degree of the individual;
- Selection: the individual chromosomes are selected for reproduction according to the elitist selection: the individual having the best fitness is chosen;
- Crossover: new populations of chromosomes are generated using the algorithm of recombination illustrated in the next paragraph and studied in [9].
- Mutation: a perturbation applied to the individuals by reversing the path between two positions.

IV.3 The algorithm of crossover

The KFP crossover is an operator of recombination where we introduced a heuristics based on a principle similar to that of a selection by roulette. It is an operator of specific crossover in the problem of the TSP (Travelling Salesman Problem).

Elaborated by Karoly F. Pale in 1993 [10], this operator creates an individual son by beginning with a city chosen at random, then by including the following cities according to a probability defined according to the distance between this city and the following city in each of both parents.

This technique produces on average good quality individuals rather quickly. Besides it introduces a disturbing factor in this operator of crossover added to the effect of the mutation operator.

Algorithm 2

c :current city

a : Position next city in parent1

b : Position next city in parent1

pA: Probability

pB: Probability

begin

files[0] <- *c*

for (*i* = 1 to *n*) **do**

a ← *position(c, parent1)+1*

b ← *position(c, parent2)+1*

while (*parent1*[*a*] ≠ *files*) *do a* ← (*a+1*)%*n*

while (*parent2*[*b*] ≠ *files*) *do b* ← (*b+1*)%*n*

```

    pA ← ωc.parent1[a]
    pB ← ωc.parent2[b]
    if pA > pB then fils[i] ← parent1[a]
    else fils[i] ← parent2[b]
end for
end

```

At the first solution we adopted this algorithm; as it's shown below in the algorithm 3 the probability (p_A and p_B) are calculated according to the distance:

```

d_a = Distance (iCurrentGnum, a);
d_b = Distance (iCurrentGnum, b);
p_a = d_a * (rand () / (RAND_MAX + 1.0));
p_b = d_b * (rand () / (RAND_MAX + 1.0));

```

Where:

d_a : distance between the current position in the individual son and the gene a in the first parent.

d_b: distance between the current position in the individual son and the gene b in the second parent.

The individuals correspond in our work to the possible paths connecting the artificial beacons which mark the configuration space of the mobile robot. These beacons form the genes of the individual. The genes are coded by means of numbers (integer type). The elaborated algorithm of crossover is the following algorithm [1][6]:

Algorithm 3

Procedure CrossoverKFP (Individual &I1, Individual &I2, Individual &I_temp)

variable

i, iCurrentGnum, iCurrentPos, a, b: integer

d_a, d_b, p_a, p_b: float

Begin

For (i=0 to NbG) do

I_temp.Amers[i].num = -1

End for

iCurrentGnum ← NbG * (rand () / (RAND_MAX + 1.0))

I_temp.Amers[0].num ← iCurrentGnum

For (iCurrentPos=1 to NbG) do

{look for the next gene after iCurrentAmers}

a ← I1.FindGenePosition (iCurrentGnum);

b ← I2.FindGenePosition (iCurrentGnum);

{ gene from parent1 I1: if it was visited we shouldn't take it }

repeat

a = (a + 1) mod NbG

until (I_temp.FindGenePosition(I1.Amers[a].num) = -1)

a ← I1.Amers[a].num;

{ gene from parent2 I2: if it was visited we shouldn't take it }

repeat

b = (b + 1) mod NbG

until (I_temp.FindGenePosition(I2.Amers[b].num) = -1)

b = I2.Amers[b].num

{ calcul the distances => probabilities related to a and b }

d_a = Distance (iCurrentGnum, a)

d_b = Distance (iCurrentGnum, b)

p_a = d_a * (rand () / (RAND_MAX + 1.0))

p_b = d_b * (rand () / (RAND_MAX + 1.0))

if (p_a < p_b) then

iCurrentGnum ← a

else

iCurrentGnum ← b

end if

```

    I_temp.Amers[iCurrentPos].num ← iCurrentGnum
    I_temp.Amers[iCurrentPos].AmersPosX ← X[iCurrentGnum]
    I_temp.Amers[iCurrentPos].AmersPosY ← Y[iCurrentGnum]
End for
{evaluation of the individual son I_temp}
I_temp.Eval()
End {CrossoverKFP}

```

IV.4 Proposed algorithm based on KFP crossover and Neighbourhood method

The Neighbourhood meta-heuristic was used to resolve the defined CSP in the section II.3.2. In [10] the word meta-heuristic was coined by GLOVER [11] and its meaning has been changing ever since. According to the original definition, meta-heuristics are methods that combine local improvement procedures and higher level strategies to create a process capable of escaping from local optima and performing a robust search for a solution space [12]. Nowadays, meta-heuristics can be seen as intelligent strategies to design or improve heuristics procedures with a high performance. Generally it combines constructive methods, local search methods, concepts that come from Artificial Intelligence, biological evolution and statistics methods [13]. In this paper we have analyzed the hybridization of the CSP resolving problem and Genetic algorithms as representative methods.

Before the description of the solution based on neighbourhood method, we will define this method as following:

Definition: The methods of neighbourhood are based on the notion of neighbourhood. First of all, we will introduce this fundamental notion as it's defined in [11]:

Let X the set of the admissible configuration of a problem:

- The neighbourhood is an application $N: X \rightarrow 2^X$;
- The mechanism of the neighbourhood exploration is a procedure that defines how to get a configuration $s' \in N(s)$ from a configuration $s \in X$. A configuration s is a local optima (minima) of the neighbourhood N if $f(s) \leq f(s') \forall s' \in N(s)$ where f is the objective function.

A typical method of neighbourhood begins with an initial configuration, and realizes then an iterative process which consists in replacing the current configuration by one of its neighbours in the objective function. This process stops and returns the best configuration found when the ending condition is realized. This condition concerns generally a limit for the number of iterations or an objective to realize.

We have opted for the method of neighbourhood to look for the gene which verifies the temporal constraint this is illustrated in the algorithm 4.

Algorithm 4

Procedure CSPverif(int T, Individual I)

Variable

l,m,n : int
d,D : float

begin

```

for (i=0 to NbG) do
    d=I.Distance(i,i+1)
    {calcul the position coordinates that respecte time window T}
    x=I.Amers[i].AmersPosX + vl*cos(a)*T;
    y=I.Amers[i].AmersPosY+ vl*sin(a)*T
    a=a+va*T
    D=sqrt(((x-I.Amers[i].AmersPosX)*(x-I.Amers[i].AmersPosX)
            +((y-I.Amers[i].AmersPosY)*(y-I.Amers[i].AmersPosY)));
    {initialization of the gene to change with the the next gene i+1}
    m=i+1
    if (d>D) then
        {test the other neighbours for each beacons (gene): neighbourhood size = 3 voisins}
        For (j=0 to 2) do
            {function FindNeighbour(i,j) return the jth neighbour of gene i}
            l←FindNeighbour(i,j)
            n←I.FindGPos(l)
            {calcul the distance between ith gene the new neighbour}

```

```

    d ← I.Distance(i,n)
    if(d < D) then
        m = n; {temporal constraint is verified}
    end if
end for
{ mutation of individual I exchange the gene i+1 by the gene m }
I.Mutate(i+1,m)
{update d}
d ← I.Distance(i,i+1)
{update the linear velocity of the mobile robot in the new position :gene}
vl = d/T
end for
end {CSPVerif}

```

Algorithm 5

The algorithm 4 is based on neighbourhood. It will be called after every crossover to test the individual son. The crossover algorithm used, was developed and tested in the previous work [6] [3] based on KFP algorithm (algorithm 3). As results, the new crossover algorithm has two steps: first one calls KFP crossover (algorithm 3) then the second calls CSPverif illustrated in algorithm 4. This assures that the individual obtained from the process of recombination, respects the temporal constraint defined in the section 2.3.1. As result we generate a solver to the optimization problem with time window described in section 2.3.2 of this paper.

```

Procedure CorssoverCSP(int T, Individual &I1, Individual &I2,Individual &I_temp)
Begin
    CrossoverKFP ( I1, I2, I_temp)
    CSPverif(T, I_temp)
End { CorssoverCSP}

```

V. Simulation and results

The simulation of the new genetic algorithm and the different functions related to the algorithms (Algorithm 3, Algorithm 4 and algorithm 5) are implemented using C ++ under the development environment of the Microsoft Visual Studio 2010.

We assume that the mobile robot is a one particle in its configuration space. It's a horizontal plan equipped by a number of artificial beacons (n = 5). Each one represents an objective to reach which is a site to survey. The positions of the artificial beacons are represented in the Table 1.

Linear speeds and angular initials are respectively 2m / and 0.5m/. The temporal window has for value 5mn and the number of neighborhood is fixed to 3. The time window has for value T=5mn and the number of neighbors is fixed to 3.

The execution begins with the initial population described by the Table 2. The evolution of the various generations in this case is described by the tables 3, 4, 5. According to the values obtained in the final generation (table 5), we can deduct that the solution is the path coded by the successive positions 04231 having a length of 12.6084.

The paths yielded by the hybrid algorithm are represented in the figure Fig.3. The evolution of the individuals resulting from every generation is illustrated by the figure Fig.4. The figure Fig.5 represents the path tracking solution of the generated algorithm which is encoded by the integer sequence: 04321.

Table 1 Coordinates' positions of the artificial beacons

artificial beacon Code	0	1	2	3	4
Axis X	7.0	9.0	5.0	3.0	5.0
Axis Y	8.0	1.0	2.0	4.0	6.0

Table 2 1st generation

Individual code	04321	01423	02413	21430	30214
fitness	12.6084	20.5117	23.4359	19.0115	22.5076

Table 3 2nd generation

Individual code	04321	01432	02413	21430	30214
fitness	12.6084	19.3401	23.4359	19.0115	22.5076

Table 4 3rd generation

Individual code	04321	01342	02413	21430	30214
fitness	12.6084	20.8167	23.4359	19.0115	22.5076

Table 5 Final generation

Individual code	04321	01234	02413	21430	30214
fitness	12.6084	18.2316	23.4359	19.0115	22.5076

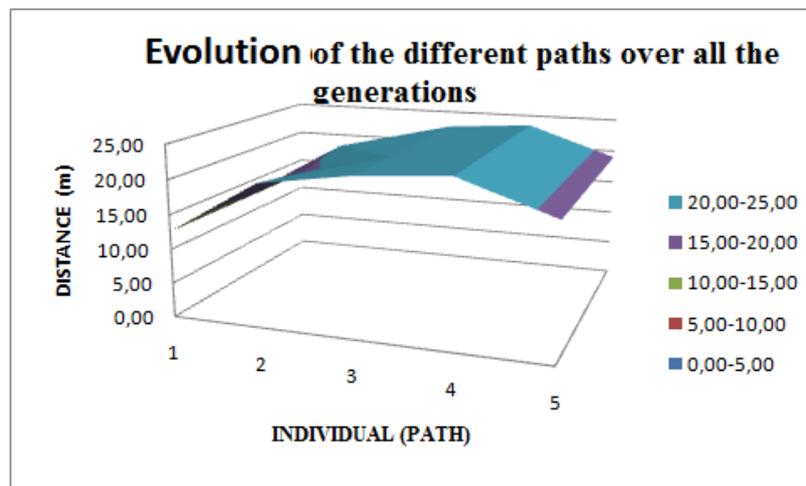


Figure 3: Paths yielded of the new genetic algorithm combining geneticalgorithms and CSPtechniques

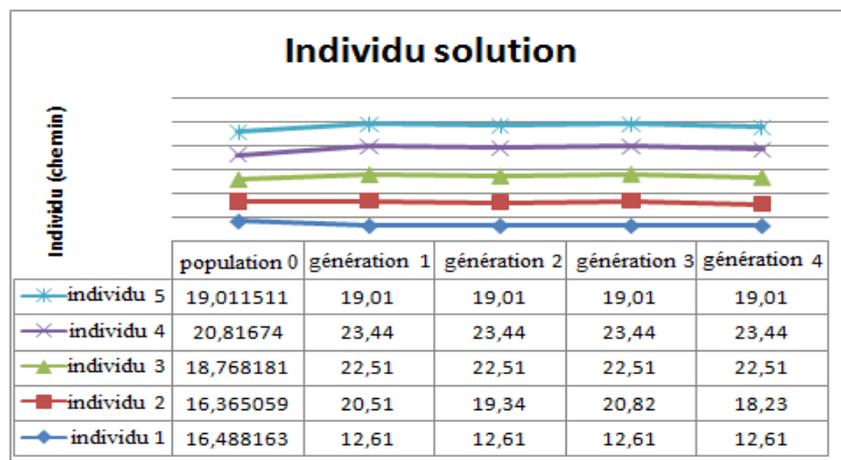


Figure 4: Evolution of different individuals of the hybrid algorithm combining geneticalgorithms and CSPtechniques

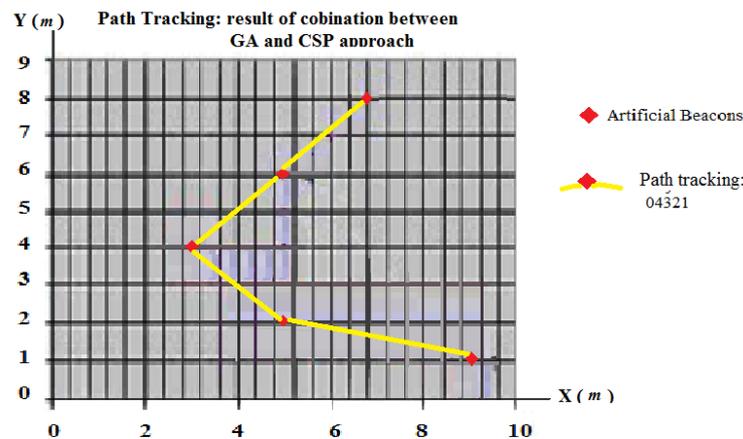


Figure 5: Path tracking: result of the hybrid algorithm combining genetic algorithms and CSP techniques

VI. Conclusion

The optimization of the mobile robot track with time window is a combinatorial optimization type that can be modelled as constraint satisfaction problems. For this reason we have used hybridization between the genetic algorithms and the CSP techniques to have a solver to the mentioned problematic.

Further than the hybridization we have used a meta-heuristics of neighbourhood method which help us to resolve the defined CSP. This method based on a set of neighbours visited by the mobile robot to keep the one going with the purpose of the developed solver. The cardinal of this set has an impact on the results of the obtained algorithm. At the future work we intend to study this outcome by scheming the ratio of the hybridization for different cardinals of the neighbours' set.

REFERENCES

- [1] H. Tijani, J. Knani, Optimization of the mobile robot's track using Genetic algorithm. In *10th International conference on Sciences and Techniques of Automatic control & computer engineering – STA'2009*, Hammamet-Tunisia, 2009.
- [2] D.E Smith, (2004). Choosing objectives in over-subscription planning, *Proc. the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*, Whistler, British Columbia, Canada, June 3-7 2004, 393-401.
- [3] S. Essayie, *Politique de sélection dans les algorithmes génétiques – Application au Voyageur de commerce*, Université de Havre, France, 2010.
- [4] E. Chantry, *Planification de Mission pour un Véhicule Aérien Autonome*, L'École Nationale Supérieure de l'Aéronautique et de l'Espace, France, 2006.
- [5] Y. Lassoued, *Etude de paramétrage d'un algorithme d'optimisation hybride*, France, 2008.
- [6] H. Tlijani, K.N. M'sirdi, J. Knani, Optimization of the mobile robot's Path Tracking: Combination of Genetic algorithm, Artificial Potential Field and Backtracking Algorithm, *Proc. International Conference on Communications, Computing and Control Applications – CCCA'11*, Tunisia, 2011, 97-102.
- [7] Betke, Gurvits, Mobile robot localisation using landmarks, *Proc. the IEEE, International conference on robotics and automation*, 2, 1994, 135-142
- [8] C. Pradalier, *Conception d'un système de localisation pour un robot mobile: utilisation d'un télémètre laser et placement des amers artificiels dans l'environnement*, Institut National Polytechnique de Grenoble, France, 2001.
- [9] H. Tlijani, J. Knani, N. K. M'sirdi, Optimization of a mobile robot's path tracking with time windows using Hybridization of genetic algorithms and CSP approach, *International Conference on METAheuristics and Natural Inspired Computing – META'12, Sousse, Tunisia, 27-31 October 2012*.
- [10] K.F. Pál, *Genetic algorithms for the travelling salesman problem based on a heuristic crossover operation*, Institute of Nuclear Research of the Hungarian, Academy of Sciences. Debrecen, Hungary, 1993.
- [11] F. Glover, W. FRED, A. GARY, *Handbook of Metaheuristics* (Kochenderberger, Eds, Kluwer Academic Publishers. Dordrecht, The Netherlands, 2003).
- [12] H. Jin-Kao, *Metaheuristics for combinatorial optimization and constrained solving*, University of Montpellier II, France, January 1998.
- [13] P. H. Mills, *Extensions to Guided Local Search*, doctoral diss., Department of Computer Science University of Essex, France, 2002.