The CRONE toolbox for Matlab: Fractional Path Planning Design in Robotics

P. Melchior, B. Orsoni, O. Lavialle and A. Oustaloup

Abstract:
Interactive communication between robots and humans is presented through a human interface design for path planning and through the use of fractional potential to take into account danger of obstacles. A pedagogic programming is also presented for both users and developers within the "Fractional Path Planning Design" module of the CRONE toolbox. Different path planning methods, which use fractional map of danger are included in the toolbox, especially A* algorithm and fast marching technique. With both methods, dangers of obstacles are gradually taken into account to anticipate collision and also to provide smoother trajectories. The fast marching technique provides systematically convex map with a unique minimum: all the trajectories can be evaluated in real time, whatever the start point.

Keywords: Matlab Toolbox, Fractional Calculus, Robotics, Computer Assisted Design, Path Planning, Danger Modelization, A* Algorithm, Fast Marching.

I. INTRODUCTION

1.1. Fractional differentiation in robotic

Fractional differentiation has been used in automatic for academic and industrial applications [Ous95][Ous99]: robust control theory with the CRONE concept, system identification using fractional model, vibration isolation, signal and image processing. A research axis aiming at introducing fractional differentiation in robotics and mechatronics has also been developed in the "Laboratoire d'Automatique et de Productique" of Bordeaux [Ous96][Ous97][Mel99][Ors00]. In path planning design, a new approach which uses Weyl fractional potential has been developed. The trajectories provided depends on dangers of obstacles and the smooth curvature of the trajectories is well adapted for path tracking design. In order to put forward the practical interest of using fractional integration design, interactive graphic interfaces have been developed and are now available in the "CRONE toolbox" for Matlab.

1.2. Fractional path planning design

In path planning design, potential fields can introduce force constraints to ensure curvature continuity of trajectories to facilitate path planning design. It is also expected in trajectory planning for mobile robots, that the danger associate to obstacles smoothly modify the trajectory. In the approach we use, the fractional order of differentiation is used as a risk coefficient for each obstacle, and determines the potential map created by the various obstacles. The parametric thrust of fractional potentials permits danger modelization generated by obstacles, allowing a continuous variation without modification of the fictive geometric distribution of charge.

The environment modelization of the danger encountered by the robot is available in the module "Fractional Path Planning Design" of the CRONE toolbox. Moreover, different path planning methods, which use fractional potential, have been programmed in the module. They are described in section 4.

1.3. The CRONE toolbox

Various modules of the CRONE toolbox have been developed gradually since the beginning of the nineties with many publications and patents in 1993 and 1994 [APP94][Ous95]. Each module deals with a specific application theme of fractional differentiation. Some modules were initially programmed in Turbo Pascal. These modules have been rewritten for Matlab v5.3 under Windows [Mel99][Ous00].

Different motivations have lead to that choice. First, solving problems in Matlab is generally much quicker than programming in a high-level language such as C or Fortran. Second, most university and industrial research and development laboratories use Matlab, which is an international standard in automatic control. These properties result from the relationship between matrices, structures, objects, functions and toolboxes in the Matlab environment. The CRONE toolbox had to integrate these properties and an original functional ergonomics have been designed. The CRONE toolbox has also evolved into an interactive interface for general scientific and technical computation and visualization described in section 2.

1.4. Programming methodology

The CRONE toolbox programming methodology uses structures and OOP as they are available since version 5 of Matlab. First, data structures are a way of grouping together information that is logically related. Structures are thus useful to condense information, which eases data exchange and backup. Second, the main advantage of object oriented methods is that they hide complexity, allow development of transparent and reusable code, and can generally be made portable.

1.5. Content

The use of structured data and object in the CRONE toolbox is detailed in section 2, which also describes the Human Interface design of the CRONE toolbox. In sections 3, the Fractional Path Planning Design module of the CRONE toolbox for Matlab has been presented.
toolbox is presented including an original modelization of the danger encountered by a mobile robot in a given environment.

Other modules are available: through this toolbox we aim to achieve our high scientific priority to make our research results on fractional mathematics widely available. The modules are developed with the help of the French CNRS and the automobile industry, were chosen to enhance collaboration with any industry concerning fractional computation.

II. HUMAN INTERFACE DESIGN

II.1. Object-oriented programming

Object-oriented language, available in Matlab since version 5, provides means for modeling. Object-oriented language has originally been developed for modelization and is adapted for a robotic application. The interpretation of the environment and phenomena are provided to the robot in a more intelligible way through a OOP language.

Object-Oriented programming in the CRONE toolbox is used first to obtain figures and windows with the same aspect without passing graphic parameters. Graphic parameters are defined one for all in the main program of the toolbox, by setting reference values in the root objet. All instances of this object are considered as descendant. Thus, graphic objects of the CRONE toolbox inherit of the reference parameters whenever they are defined (fig. 1, 2). The use of inheritance permits to reuse definition code of the graphic object, and gives the flexibility to slightly modify it, if the old code does not do exactly what needed for the new graphic object.

An other way of using Object-Oriented Programming is the various high-level graphing routines, available in Matlab. Thus, the definition of dialogue box windows or pull-down menus, require only few code lines in the CRONE toolbox (fig. 3 and 4).

```matlab
set(0,'DefaultFigureColor',[.8 .8 .8]...,'DefaultFigureUnits','normalized';...,'DefaultFigureNumberTitle','off';...,'DefaultFigureResize','on';...,'DefaultFigurePointer','Arrow';...,'DefaultFigureMenu','none';...,'DefaultControlBgcolor',[180 180 180]/255,...,'DefaultControlUnits','normalized');
```

Parameters inherited and reusable function, allow a structured programming. A quick evolution of each module is thus possible by including few code lines in the graphic interface, by developing new high-level function or method, and by using existing functions, and structure.

II.2. Folders hierarchy

Each module of the CRONE toolbox inherits the same structure (fig 5).

The root folder (graphic interface level) of the module, contains structured and user-friendly pull-down menu definitions. The method folder (algorithmic level), contains high level functions which describe methods, namely dialogue box definitions, function appeals and data shaping. The function folder, (functional level), contains various functions that an advanced user needs. The funfun folder contains small subfonctions used by a function from the function folder for optimization or intermediate calculus. Subfunction definition within a function can be used if the subfunction is very specific and can not be reused.

A documentations folder and a Data folder are usually used in order to store data and to include documents for the description of methods or functions. Specific folders can also be used for a module: in the "Fractional Path Planning Design” module, a Bmp folder is defined.
II.3. Modules (user interface level)

Matlab graphic functions allow efficient implementation of various pull-down menus. User interface definition is the unique purpose of the main function in the root folder of the module: nor global, nor local functional variables are defined at this level. Links between the graphic interface and functions are programmed through the callback routines, which is one property of user interface menus and controls.

![Image](image.png)

**Fig. 6. Visualization command "Modified energy", within the User Interface of the "Fractional Path Planning Design" module in the CRONE toolbox**

```matlab
pre = uimenu(Label,'Fractional potentials');
ui menu(pre,Label,'Grunwald','Callback','f_Visua(6)');
ui menu(pre,Label,'Weyl','Callback','f_Visua(7)');
```

**Fig. 7. Pull-down menu definition**

A parameter selection can appear within the callback instruction. For example, polynomial interpolations are grouped in a method and the order parameter occurs in the callback instruction. In the example (Fig. 7), the method f_visua includes different visualization operations of the same type.

II.4. Methods

Users reach the method level through the graphic interface by a simple evolution in a pull-down menu and a click (Fig. 6).

Variables defined at this level can be local or global. Global variables characterize this level as they can not be used somewhere else. They ensure parametric exchange through the various methods without appearing as parameters in the function definitions. So, input and output variables are not required in the callback parameter within the main function of the root folder of the module. Thus, graphic specifications at the users interface level, and algorithm at the algorithmic level, are put forward.

II.5. Functions (functional level)

Functions defined at this level must be available directly from Matlab. They must include a complete and clear inline help and must be usable and reusable.

For choosing functions, required by the methods, the module design, in the CRONE toolbox, can not impose a methodology but allows the designer to use both descending and ascending analysis. As interpretation problems must be quickly solved during functional analysis, existing functions must be reused and reusable functions must be identified.

II.6. Programming Design

Programming design leads both users and developers.

For developers, the main objective is the design of a friendly user interface. Programming design leads first the introduction of new methods by optimizing the code required: this reduces efforts and delays. The user interface is thus programmed in the same way to have the same functional ergonomics of user interface controls.

For users, the main requirement is not only the technical need but also the easy of use of the methodology. Programming design ensures the same aspect for graphic interfaces. Few types of user interface controls must be used to reduce and simplify the number of procedures that the user should know. Functional ergonomic design of user interface and delay reduction are thus compatible. They are both provided by the programming design.

III. "FRACTIONAL PATH PLANNING" MODULE

III.1. Module description

Various path planning methods are available in the "Fractional Path Planning Design" module, especially methods which using fractional potentials.

Obstacles of the robot environment are designed in the menu "parameters" with their respective danger coefficients, which correspond to fraction orders of integration. The menu "Maps", computes various distance or danger maps, which provide information required for the trajectory determination. The menu "Trajectories", provides trajectories from gradient descent or heuristically graph development. Different visualizations are also available.

Other software tools currently available cannot be used for systems with fractional derivatives (i.e. non integer or complex order derivatives). The "Fractional Calculus" module includes all algorithms, which allow the use of fractional or complex order derivation. Functions of this module are thus used within the "Fractional Path Planning Design" module.

III.2. Fractional integration

The fractional integration concept was born in the 19th century with Cauchy, Laplace, Liouville Abel, and Riemann. The integration order is no longer limited to being an integer. Riemann’s definition is [Mil93][Sam93]:

$$
^{c}D_{t}^{-n}f(t) = \frac{1}{\Gamma(n)} \int_{c}^{t} (t - \xi)^{n-1} f(\xi) \, d\xi ,
$$

where $n$ is the fractional integration order. Unfortunately, for historical reasons, the word fractional is used in the literature as opposed to integer. However, it is understood as meaning any real, imaginary or complex number.

Weyl's definition can also be used to have an integral
reference at $\infty$ [Mil93][Sam93]. Weyl’s definition is:

$$W^{-n}f(t) = \frac{1}{\Gamma(n)} \int_{t}^{\infty} (\xi-t)^{n-1} f(\xi) \, d\xi.$$  \hfill (2)

For example, considering the function:

$$f_\alpha(r) = \frac{1}{r^\alpha},$$  \hfill (3)

for which the laplace transform is not defined when $\alpha > 1$. However, Weyl fractional integral of the function (3) is defined:

$$\forall r > 0, \quad W^{-n}r^{-\alpha} = \frac{\Gamma(\alpha-n)}{\Gamma(\alpha)} r^{n-\alpha},$$  \hfill (4)

$$\forall \alpha, \forall n, \text{ such as } \Re(\alpha) > \Re(n) > 0.$$

The fractional differentiation of a function can be evaluated using Grünwald’s definition [Gru67]:

$$D^n f(t) = \frac{1}{h^n} \sum_{k=0}^{\infty} (-1)^k \binom{n}{k} a_k(n) f(t-kh).$$  \hfill (5)

This becomes the same as the Riemann-Liouville fractional integral when the sampling interval tends towards zero [Sam93]. A negative real part for the fractional order of differentiation $n$ is chosen for (5), so the fractional integral (1) can be computed.

III.3. Danger modelization through fractional potential

Path planning design is the elaboration of a strategy to define a trajectory, which will reach a target avoiding obstacles [Vol90].

Potential field methods for path planning are often an alternative to graph searching techniques [Bar91][Hwa92][Pru96]. They introduce force constraints for practical speed control. The potential field concept considers the robot as a charged particle moving under the influence of repulsion potentials from obstacles, and attraction potentials of the target. The potential value gives a danger level at each point of the environment [Kha86][Vol90]. The potential gradient gives the norm and direction of the force field. In most cases, the force norm is not use; but the direction of the force provides the most appropriate heading for the robot to take. As the trajectory depends only on the local environment of the robot, it is thus available in real-time. The smoothness of the curve obtained with potential field methods, makes practical steering and speed control possible. However, the robot may not find its way through a narrow path. The robot may also be trapped in local minima as the algorithm always indicates the deepest point of potential cups.

The parametric thrift of fractional potentials permits to take into account the danger of obstacles through a risk coefficient, which corresponds to a fractional order of integration. The Coulombian potential of an isolated charge and an infinite segment are included in fractional potential. Moreover, variation of the fractional order generates smooth variation of potential, without requiring design of geometric charge distribution.

III.4. Fractional potentials

Using fractional integration, the charge influence should be continuously and gradually controlled without the need for modification of fictive geometric distribution [Ors00].

As the potential field is the electric field circulation, the fractional potential field is then defined as the fractional integral of the electric field generated by an isolated charge. In an electrostatic context, energy and potential field of a charge distribution are evaluated from $r = \infty$ to $r$. Riemann’s definition is not appropriate for such an integral reference. Weyl’s definition considers a reference at infinity and is therefore chosen:

$$V_n(r) = W^n(E(r)e_r),$$  \hfill (6)

As $E_0(r) = \frac{q}{4\pi\varepsilon_0 r^2} e_r,$

$$\forall n \in [0,2], \quad V_n(r) = \frac{q}{4\pi\varepsilon_0 r^2} r^{2-n}.$$  \hfill (8)

The potential is also normalized to correspond to a relative danger level. The normalized potential $U_n(r)$ must be null (0% of danger) at and greater than a finite distance considered as the maximum distance of influence $r_{\text{max}}$. It must be unity (100% of danger) at each obstacle to forbid collision. As a singularity appears when $r = 0$, a finite minimum distance $r_{\text{min}}$ is chosen. The normalized potential is thus:

$$\forall n \in [0,2], \quad U_n(r) = \frac{V_n(r) - V_n(r_{\text{max}})}{V_n(r_{\text{min}}) - V_n(r_{\text{max}})}.$$  \hfill (9)

Substituting equation (8) in definition (9), the analytic expression of the Weyl’s normalized fractional potential is also deduced:

$$\forall n \in [0,2], \quad U_n(r) = \frac{r^{n-2} - r_{\text{min}}^{n-2}}{r_{\text{max}}^{n-2} - r_{\text{min}}^{n-2}}.$$  \hfill (10)

A continuity extension to $n = 2$ is defined:

$$\forall r \in [r_{\text{min}}, r_{\text{max}}], \quad U_2(r) = \frac{\ln(r_{\text{max}}) - \ln(r)}{\ln(r_{\text{max}}) - \ln(r_{\text{min}})}.$$  \hfill (11)

and also for $n > 2$:

$$\forall n > 2, \quad U_n(r) = \frac{r^{n-2} - r_{\text{min}}^{n-2}}{r_{\text{max}}^{n-2} - r_{\text{min}}^{n-2}}.$$  \hfill (12)

The charge no longer effects the normalized potential, but the potential shape is continuously controlled by the fractional integration order. The potential shape only depends on the fractional integration order that corresponds to the danger of the obstacle. The more dangerous the obstacle, the greater the order of integration; also, the greater the influence area, the greater the potential for a given distance.

Weyl’s fractional potential is thus a compact, easy to use, expression. No integration is required for its computation, and the potential shape only depends on the fractional integration order that corresponds to the danger of the obstacle.
IV. FRACTIONAL MAP, FRACTIONAL ROAD

The fractional potential field map (fractional map) indicates a danger level for each point of the free space.

For each obstacle the potential is evaluated from the nearest charge [Lat96], to avoid area integration. Using, the distance map for each obstacle can now be converted into a potential map. The sum of all the maps can define the fractional potential map of the environment, but the obstacle danger level would not be normalized. For each point of the free space, the maximum value of all the potential maps is thus chosen, thus providing the global fractional map. So, the potential values are normalized. Also, local maximum values of potentials are avoided, but this leads to discontinuity of the force field directions.

A graphical representation of the fractional potential map is given in figure 9. The set of free space points, which have a less than threshold value, defines the fractional road map (figure 10). Keeping the robot on the fractional road ensures the respect of the danger level.

The fractional road map includes a security distance around the obstacles. This distance depends on the danger level of obstacles: the greater the danger, the greater the security distance.

V. A* ALGORITHM USING FRACTIONAL POTENTIAL

The Fermat principle states that a ray always chooses a trajectory \( \Gamma \), that minimizes the optical path length [Die72]:

\[
\int n_r(x, y, z) \, ds ,
\]

where \( n_r \) is the refractive index.

The path must be a local extremum, usually a minimum but in rare cases a maximum. Thus, if the fractional potential field can be the refractive index, the minimum energy trajectory (energy being the circulation of the refraction index) would be the time-optimal trajectory.

A weighting factor in the determinist evaluation function allows both energy and length of the provided trajectories to be considered. Thus, this evaluation function allows smooth passage from minimum length to minimum energy trajectories. The trajectory length is then evaluated with a third dimension given by the fractional potential. The criterion considered in the determinist function of the evaluation function of the A* algorithm [Nil69][Nil71] [Far97] is thus:

\[
g(N) = g(N-1) + \sqrt{d(N,N-1)^2 + \lambda \ p_N^2} ,
\]

where \( d(N,N-1) \) is the 2D displacement between two successive nodes and \( p_N \) the potential in the new position considered.

The flying distance between the current node and the target is also chosen as the heuristic function:

\[
h(N) = d(N, N_t) .
\]

Using such an heuristic function ensures that the minorance property is true and the A* algorithm converges systematically to the optimal solution whenever the target is reachable.

For the following examples, risk coefficients are fixed arbitrarily at 1.2 for the square, 2.7 for the small rounded rectangle and 4.5 for the oval obstacle; maximum distance of influence and minimum distance of evaluation are \( r_{max} = 20 \) and \( r_{min} = 1 \).
cases toward an optimal solution [Far97]. The weighting factor between length and energy provides a continuous modification of the trajectories. Danger of obstacles is gradually taken into account to anticipate collision and also to provide smoother trajectories (figures 11-14).

The equation to solve, deduced from discretization of the Eikonal equation (16), is:

$$
\left[ \max \left( u - U_{i-1}, u - U_{i+1}, 0 \right) \right]^2 + $$
$$
\left[ \max \left( u - U_{i-1}, u - U_{i+1}, 0 \right) \right]^2 = F_{i,j}^2,
$$

(17)

The fast marching technique is available in the "Fractional Path Planning Design" module. The potential is particularized considering both displacement and danger:

$$
F_{i,j} = 1 + \lambda P_{i,j},
$$

(18)

where $\lambda$ is a weighting factor and $P_{i,j}$ the fractional potential which correspond to a danger.

A modified gradient is also used to deduce the trajectory from the convex maps provided by fast marching: intermediate points are memorized without being rounded. For the following examples, risk coefficients are fixed arbitrarily at 1.2 for the square, 2.7 for the small rounded rectangle and 4.5 for the oval obstacle; maximal distance of influence and minimal distance of evaluation are respectively $r_{\text{max}} = 50$ and $r_{\text{min}} = 1$; the maximal admissible danger is 80%. The minimum length trajectory is given in figure 15 by choosing $\lambda = 0$. It is a true approximation of the continuous trajectory. For $\lambda = 3$ or $\lambda = 10$ the criterion combines energy (danger of the trajectory) and length. The trajectories obtained are given in figures 16 and 17. As for $A^*$ algorithm, it is almost minimum length; it is always smooth curves. The minimum energy trajectory is given in figure 18 by choosing $\lambda \rightarrow \infty$.

VI. FAST MARCHING USING FRACTIONAL POTENTIAL

Fast Marching Methods [Set96] and Level Set Methods [Osh88] are numerical techniques, which solve the Eikonal equation:

$$
|\nabla U(M)| = F(M).
$$

(16)

The advanced front moving with speed $F(M)$, can thus be computed and the fast marching technique provides an extremely fast computation. This technique has a wide range of applications. It and can be used in image processing [Lav99] and also in path planning design to provide the robot a distance or time map from a target.

Trajectories provided by $A^*$ algorithm for a criterion which combines energy and length

(white: free space; grey: dangerous area; black: obstacles)

The 16 nearest neighbors of the node to be developed are considered. However, the minimum length trajectory provide by the $A^*$ algorithm for $\lambda = 0$, is not the discrete approximation. The error, even small, leads to jumpy steering. Moreover, the error is not necessarily reduced when the pixel accuracy is increased. As danger and length are decorrelated, error is reduced when they are both considered in the criterion. Thus considering both length and danger ensures smoother and more accurate trajectories.

Even if the heuristic leads the graph development, the trajectory can not be evaluated in real time by the robot. Moreover, for different start points, the computation can not necessarily be reused. A complete convex map computation can thus be preferred that leads to consider fast marching technique.
map with a unique minimum. The solutions obtained are the true approximations of the continuous and optimal trajectories. Furthermore, the complete convex map can almost be computed in real time. Anyway, the convex map can be transferred one for all to the robot and whatever the start point, all the trajectories can be evaluated in real time.

VII. CONCLUSION

Interactive communication between robots and humans has been presented through a human interface design for path planning and by using fractional potential to take into account danger of obstacles.

The "Fractional Path Planning Design" module of the CRONE toolbox uses object-oriented language, available in Matlab since version 5. This approach is adapted for a robotic application. The interpretation of the environment and phenomena are provided to the robot in a more intelligible way. Furthermore, parameters inherited and reusable functions allow a structured programming and a quick evolution of each module. A pedagogic programming has also been designed for both users and developers.

In path planning design, the Weyl fractional potential is used to compute fractional maps, which provide a danger level for each point of the free space. The fractional integration order controls the potential shape smoothly to characterize various dangers generated by obstacles. Fractional potential can also be used in a heuristically guided search or fast marching technique. The A* algorithm converges systematically toward an optimal solution. The weighting factor between length and energy provides a continuous modification of the trajectories. Danger of obstacles is gradually taken into account to anticipate collision and also to provide smoother trajectories. The fast marching technique leads systematically to convex map with a unique minimum. The solutions obtained are the true approximations of the continuous and optimal trajectories. Moreover, the convex map can be transferred one for all to the robot and whatever the start point, all the trajectories can be evaluated in real time.

Minimum curvature radius, maximal speed and maximum acceleration have been used in previous fractional potential definitions [Ous95][Ous96]. Such constraints must now be used with the analytic Weyl fractional potential, and with the A* algorithm or fast marching technique. Dynamic constraints, including actuator band-pass and bending modes, must also be introduced.

Fractional potential gives a free space representation that can provide an integrated method of path planning and tracking design. The practical method, under construction, should determine the time optimal trajectory with obstacles of any shape and considering actuator limitations.

VIII. REFERENCES

[APP94] APP (Agence pour la Protection des Programmes), n°93.30.006.00 (IDDN.FR.001.300060.00. R.P.1993.000.0000), le 28/07/1993 et n°94.11.015.00 (IDDN.FR.001.110015.00. R.P.1994.000.0000) - le 16/03/1994.


