ANALYZE OF A COLLISION AVOIDANCE STRATEGY FOR COOPERATIVE ROBOTS

Prof. Eng. **A. Fratu** PhD[1], Prof. Eng. **M. Dambrine** PhD[2, 3]

[1]University „Transilvania" from Braşov, Brasov, Romania
[2]Univ Lille Nord de France, F-59000 Lille, France
[3]UVHC, LAMIH, F-59313 Valenciennes, France

REZUMAT. **Aceasta lucrare prezinta o analiza a unei strategii noi de evitare a coliziunii robotilor cooperativi. Fiecare robot fizic actioneaza total independent, comunicand cu prototipul virtual corespunzator si imitand omologul sau virtual. Fiecare robot fizic reproduce comportamentul prototipului sau virtual. Estimarea actiunilor fara coliziuni, folosind robotii virtuali, si munca in colaborare a robotilor fizici care imita prototipurile lor virtuale sunt ideile originala ale lucrarii. Noi am testat prezenta strategie pe cateva scenarii de simulare care implica doi roboti virtuali, estimand actiunile fara coliziuni in timpul sarcinilor de cooperare.**

Cuvinte cheie: Roboti virtuali, Roboti cooperativi, Evitarea reciproca a coliziuni

ABSTRACT. **This paper, presents an analyze of a new collision avoidance strategy for cooperative robots. Each physical robot acts fully independently, communicating with corresponding virtual prototype and imitating her virtual homologue. Each physical robot reproduces the behaviour of her virtual prototype. The estimation of the collision-free actions of the virtual robots and the collaborative work of the physical robots who imitate there virtual prototypes, are the original ideas. We tested the present strategy on several simulation scenarios, involving two virtual robots and estimating collision-free actions, during of the cooperative tasks.**

Keywords: Virtual robots, Cooperative robots, Reciprocal collision avoidance

1. INTRODUCTION

Collaborative robots, we see being deployed nowadays in research or industry, are permanently in danger to be in collision. Therefore installations with multiple robots in real world, such as collaborative work and maintenance the good state of the line production, require collision avoidance methods, which take into account the mutual constraints of the robots.

A key requirement for their efficient operation is good coordination and reciprocal collision avoidance.

The contact of the robot with an obstacle must bee detected and it will cause the robot to stop quickly and thereafter back off to reduce forces between the robot and environment. The problem of the contact with obstacle imposes the null velocity in the moment of the impact.

The problem of the contact detection is better analyzed on the virtual prototype in the virtual environment, where the virtual objects can be intersected and there no exist the risk to be destroyed.

The problem of the contact detection in the virtual environment on the virtual robots is important for the reason that this built-in function is proven superior to mechanical collision detection devices. Using this strategy one detects collisions in all directions, protecting not only the physical end-effectors but also the work pieces and the physical robot itself.

In the case that the real (physical) robot will imitate her virtual prototype, it has no mechanical parts which gives it higher reliability and more cost efficiency. Also, since there is no device attached to the real robot tool, one not extends the tool offset distance, which allows bigger maximum tool weight and better reorientation performance.

The problem of local collision-avoidance differs from motion planning, where the global environment of the robot is considered to be known and a complete pathway towards a goal configuration is planned at once.

So, the collision detection simply determines if two geometric objects are intersecting or not. The intersecting of two objects is possible in the virtual world, where one may predict there behavior.

The ability of predicting of the behavior of cooperative manipulators is important for several reasons: for example, in design the designers want to know whether the manipulator will be able to perform a certain typical task in a given time frame; in creating feedback control schemes, where stability is a major problem, the control engineer cannot risk a valuable piece of equipment by exposing it to untested control strategies. Therefore, a facile strategy for collision avoidance, capable of predicting the behavior of a robotic manipulator, or of a system at whole, for that matter becomes imperative.

In a real world, like collision detection, where the robots need to interact with their surrounding, it is important that the computer can simulate the interactions of the cooperative participants, with the passive or active changing environment, using virtual prototyping.

In this paper, we propose a fast method that simultaneously determines actions for two virtual robots that each may have different objectives. The actions are computed for each virtual robot and are transferred to corresponding physical robot, with a central coordination for the cooperative tasks. Yet, we prove that our method guarantees the collision-free motion for each of the robots.

We used a simplified robot model, where each virtual robot end-effectors is assumed to have a simple spherical shape moving in a three-dimensional workspace. Furthermore, we assume that each virtual robot end-effectors can moves in any direction, such that the control input of each robot is given by a three-dimensional velocity vector. Also, we assume that our algorithm is able to deduce the exact shape, position and velocity of obstacles and of the virtual robots, in the virtual environment.

The present simulation method is based on velocity approach which provides a sufficient condition for each robot to be collision-free for at least a fixed amount of time into the future.

That implies, that each robot takes into account the observed velocity of other robots in order to avoid collisions with them, and also that the robot selects its own velocity from its velocity space in which certain regions are marked as “forbidden” because of the presence of other robot.

In this paper we develop an formally analyze of a new collision avoidance strategy for a group of two collaborative robots.

2. STATE-OF-THE-ART AND POSSIBLE EXTENSIONS

The problem of collision avoidance has been extensively studied. Many approaches assume the observed obstacles to be static (i.e. non-moving), and compute an immediate action for the robot that would avert collisions with the obstacle, in many cases taking into account the robot’s kinematics and dynamics.

If the obstacles are also moving, such approaches typically repeatedly “plan again” based on new readings of the positions of the obstacles.

However, such approaches are insufficient for multi-robot location, where the robot encounters other robots that also make decisions based on their surroundings: considering them as moving obstacles neglects the fact that they react to the robot in the same way as the robot reacts to them, and inherently causes adverse actions in the motion of the robots [9].

Velocity Obstacles (VO) [1, 3] have been a successful velocity-based approach to avoid collisions with moving obstacles; they provide a sufficient and necessary condition for a robot to select velocity that avoids collisions with an obstacle moving at a known velocity.

Besides the Velocity Obstacle approach, many other methods have been proposed for collision-avoidance, navigation, and planning among moving obstacles [3]. There is also Recursive Velocity Obstacles and Common Velocity Obstacle methods.

There is also an extensive amount of literature on multi agent navigation, in which each agent navigates individually among the other agents, which are considered as obstacles, e.g. [3, 4, 11]. Most of these techniques have focused on multitude simulation. Also in these cases, the other agents are assumed to be either passively moving obstacles or static obstacles. A number of approaches follow the Velocity Obstacle concept to avoid other agents.

However, the robot may not be able to communicate with other entities and may not know their intents. It is the case of Reciprocal Velocity Obstacles(RVO) problem [9], in which robots are typically given half the responsibility of avoiding pair prudent collisions. This formulation only guarantees collision-avoidance under specific conditions, and does not provide a sufficient condition for collision-avoidance in general.

For overcomes this limitation there exist Optimal Reciprocal collision Avoidance (ORCA) that provides a sufficient condition for multiple robots to avoid collisions among one another, and thus can guarantee collision-free navigation. One distinguishes decoupled multi-agent navigation from centralized multi-agent planning.

In this paper, we present a study that provides a sufficient condition for two virtual cooperative robots to avoid collisions among one another. The virtual cooperative robots move in a complex virtual environment containing both static and moving obstacles. Our approach treats the virtual prototype of two cooperative robots who navigate while guaranteeing collision-free motion. In a future paper we transfer the trajectory of the virtual robots to physical robot in the real environment, assuming that each physical robot can perfectly imitate the movement of her virtual prototype.

3. COLLISION DETECTION

Collision detection frequently arises in various applications including virtual prototyping, dynamic simulation, interaction, navigation and motion planning. Collision detection has been exhaustively researched for more than three decades.

In this section, we present a study based on collision detection algorithm for computing all the contacts between multiple moving virtual objects in a large virtual environment. It uses the visibility reducing algorithm described in [2]. The overall algorithm is general and applicable to all environments. We also highlight many optimizations and the visibility queries used to accelerate the performance of the algorithm.

Algorithms for narrow phase can be further subdivided into efficient algorithms for convex objects and general purpose algorithms based on spatial partitioning and BVHs for polygonal models [2].

However, these algorithms often involve pre-computation and are mainly designed for rigid models.

The performance of collision detection depends on the input model complexity and the problem output, which is the number of colliding or overlapping primitives.

However, existing algorithms may not achieve interactive performance on large models consisting of thousands of triangles due to their high complexity and output of the problem. Moreover, the memory requirements of these algorithms are typically very high.

Potentially Colliding Set. We compute a Potentially Colliding Set **(**PCS) of objects that are either overlapping or are in close proximity [2]. If an object O_i does not belong to the PCS, it implies that O_i does not collide with any object in the PCS. Based on this property, we can reduce the number of virtual object pairs that need to be checked for exact collision.

This is similar to the concept of computing the potentially visible set (PVS) of primitives from a viewpoint for spatial relation. We perform visibility computations between the objects in image space to check whether they are potentially colliding or not.

Given a set *S* of objects, we test the relative visibility of an object *O* with respect to set *S* using an image space visibility query. The query checks whether any part of *O* is spatial intersected by *S*, rasterizing all the objects belonging to set *S*. The object *O* is considered fully-visible if all the fragments generated by the rasterization of *O* have a depth value less than the corresponding pixels in the frame buffer.

Many applications need to compute the exact overlapping features (e.g. triangles) for collision response. We initially compute the PCS of objects based on the algorithm highlighted above. Instead of testing each object pair in the PCS for exact overlap, we again use the visibility formulation to identify the potentially intersecting virtual regions among the objects in the PCS.

Initially, all the objects belong to the PCS. Firstly we perform reducing along each coordinate axis by using the axis aligned bounding boxes as the object's representation for collision detection.

We decompose each object into sub-objects. A sub-object can be a bounding box, a group of *k* triangles (say a constant *k*), or a single triangle. We have extended this approach to sub-object level and compute the potentially intersecting areas.

4. RECIPROCAL COLLISION AVOIDANCE

For two robot end-effectors *A* and *B*, the velocity obstacle, $VO^{\tau}_{A|B}$ (read: the velocity obstacle for *A* induced by *B* for time window τ) is the set of all relative velocities of *A* with respect to *B* that will result in a collision between *A* and *B* at some moment before time τ [9]. It is formally defined as follows.

Let be an *A* robot end-effector with radius r_A , positioned at $\boldsymbol{p}_A$ on a horizontal disc. For a configuration of two robot end-effectors *A* and *B*, the horizontal disc of radius ($r_A + r_B$) is centered at ($\boldsymbol{p}_B - \boldsymbol{p}_A$) in the Cartesian space.

Let *S* (***p***, *r*) denote an open horizontal disc of radius *r* centered at vector position ***p*** and defined by the (1).

$$S(\boldsymbol{p},r)=\{\boldsymbol{s}\in S, \|\boldsymbol{s}-\boldsymbol{p}\|\le r\} \quad (1)$$

Then the velocity obstacle is defined as:

$$VO^{\tau}_{A|B}=\{v\ |\exists t\in[0,\tau], vt\in S(\boldsymbol{p}_B-\boldsymbol{p}_A, r_A+r_B)\} \quad (2)$$

Let $\boldsymbol{v}_{tA}$ and $\boldsymbol{v}_{tB}$ be current the task (operational) velocities of the robots' end-effectors *A* and *B*, respectively. The definition of the velocity obstacle implies that if ($\boldsymbol{v}_A$-$\boldsymbol{v}_B$)∊$VO^{\tau}_{A|B}$, or equivalently if ($\boldsymbol{v}_B$ -$\boldsymbol{v}_A$) ∊$VO^{\tau}_{B|A}$, then *A* and *B* will collide at some moment before time τ if they continue moving at their current velocity.

On the other hand, if ($\boldsymbol{v}_A$-$\boldsymbol{v}_B$)¢$VO^{\tau}_{A|B}$, the two robot end-effectors *A* and *B* are guaranteed to be collision-free for at least τ time. Robot end-effector *A* will collide with robot end-effector *B* within τ time if its velocity $\boldsymbol{v}_A$ is inside $VO^{\tau}_{A|B}$ and it will be collision-free for at least τ time if its velocity is outside the velocity obstacle.

For an articulated robot arm, the robot end-effectors velocity vector is calculate as, $v=\dot{X}_t$ where $\boldsymbol{X}_t$ is the Cartesian position vector of the robot end-effector. The vector $\boldsymbol{X}_t$ can be described as a function of robot joints variables vector, $\boldsymbol{q}$:

$$X_t = \mathrm{f}(\boldsymbol{q}) \quad (3)$$

Equation (3) can be obtained easily with the help of the Denawit - Hartenberg operators.

The robot end-effectors velocity vector **v** can be obtained as:

$$v = \dot{X}_t = \boldsymbol{J}_t(\boldsymbol{q})\dot{\boldsymbol{q}} \quad (4)$$

where $\boldsymbol{J}$ is Jacobian matrix, and $\dot{\boldsymbol{q}}$, the robot joints velocities vector.

Given a trajectory that each moving robot end-effector will travel, we can determine the exact collision time.

If the path that each robot end-effector travels is not known in advance, then we can calculate a lower bound on collision time. This lower bound on collision time is calculated adaptively to speed up the performance of dynamic collision detection.

5. COLLISION DETECTION THROUGH ANIMATION OF THE VIRTUAL ROBOTS

Interactive animation of the virtual agents/objects in virtual reality systems has been a challenging problem for years.

Although many fast methods of animation have been proposed, few techniques are currently able to dynamically animate even simple virtual objects as rigid objects at interactive rates.

This paper proposes an approach that finds a balance between these two goals, enabling robust interactive animation as moving picture illustrate.

Our proposed scheme for dynamic simulation or animation, using the distance computation algorithm is an iterative process which continuously inserts and deletes the object pairs from a stack according to their approximate time to collision, as the objects move in a dynamic environment.

The end-effectors pair which has a small separation is likely to have an impact within the next few time instances, and those virtual pairs which are far apart from each other cannot possibly come to interfere with each other until certain time.

Simulation Framework. The essence of this framework is to describe each rigid object in the planning scene as a dynamical system, which is characterized by its state variables (i.e. position, orientation, linear and angular velocity). In this framework, a robot arm can be a collection of rigid bodies, subject to the influence of various forces in the workspace, and restricted by various motion constraints.

This transforms a motion planning problem into a problem of defining suitable constraints, and then simulating the rigid body dynamics of the scene with each constraint acting as a virtual force on the objects such as the collision will be avoided. Current commodity graphics hardware supports an image-space occlusion query that checks whether a primitive is visible or not.

In order to support such a query, we change the depth test to pass only if the depth of the incoming fragment is greater than or equal to the depth of the corresponding fragment in the frame buffer.

With this depth the comparison function, we used an image potentially intersecting regions query to test if a primitive is not visible when rendered against the depth buffer.

We would like to investigate different representations to support efficient runtime updates of the model's multi-resolution.

The current algorithm is designed primarily for collision detection. We used our layouts to improve the performance of view-dependent rendering, collision detection without any modification of the algorithm or runtime applications.

For real-world applications various ways to overcome this problem have been used. An extensive dissipative force, opposite to the velocity of a mass point, provides a good way to maintain the stability of the integration.

Planning scene formulation for cooperative tasks. In the cooperative tasks studies, the simulation is used to find whether it is possible to avoid the collision between a particular part of the robot arm and diverse objects in the work space and so to find one possible free path.

In this section we will describe how to render a planning scenario in the form of constraints for the constraint-based planning framework. Our visibility based on PCS computation algorithm is based on hardware visibility query which determines if a primitive is fully-visible or not.

By assuming that the geometry representing the robots and obstacles is given, as well as prescribed motion, simulated for the obstacles over time. Our system, then defines constraints that will restrict the motion of the robots to meet the design specifications, and also guide the robots to complete the planning tasks such as the collision will be avoided.

Our virtual system was implemented on a programming platform, using Delphi object-oriented programming language. We have tested our system of two robots for collision detection in the following scenes for the virtual prototyping applications:

Scene 1: individual task for each robot

Each of two robot arms is composed of rigid components that are held together by constraints. For all of the components of the robot arm, the planner must compute paths to satisfy the joint constraints do not collide with the obstacles as the conveyors and lead the end-effectors along the prescribed path.

In the scene seen in Fig. 1, two articulated robot arms, with six degrees of freedom, are used to manipulate hard objects from a conveyor belt to another conveyor belt. Each robot arm follows a path over the conveyor body while avoiding obstacles.

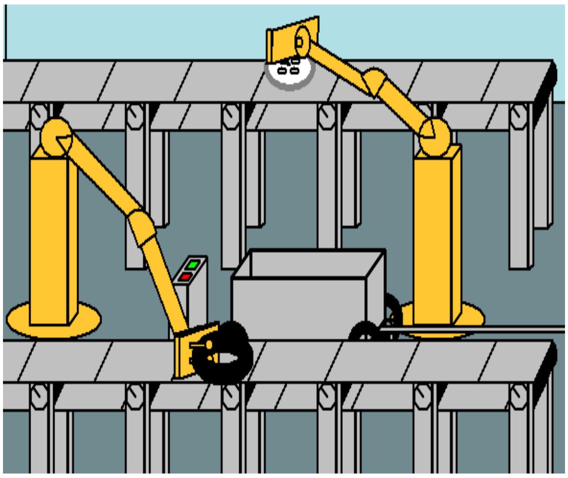

Fig. 1 Scene for individual tasks

Scene 2: cooperative mission

In our example, a second scene shown in Fig. 2, the end-effectors of the left robot and of the right robot respectively, avoid each other in a work cycle during a collaborative task. In this scene two articulated robot arms, each other with six degrees of freedom, are in cooperative mission. They are used to assembly together two rigid pieces. The robot arms avoid the moving belt to get in touch the two pieces passing them on the assembly conveyer belt.

The goal to manipulate together the same object in a cooperative mission requires both robots to maneuver around each other without colliding. The task is for each virtual robot to independently (and simultaneously) select a new velocity for itself such that both virtual robots are guaranteed to be collision-free for at least a fixed amount of time, when they would continue to move at their new velocity. As a secondary objective, the virtual robots should select their new velocity as close as possible to their preferred velocity.

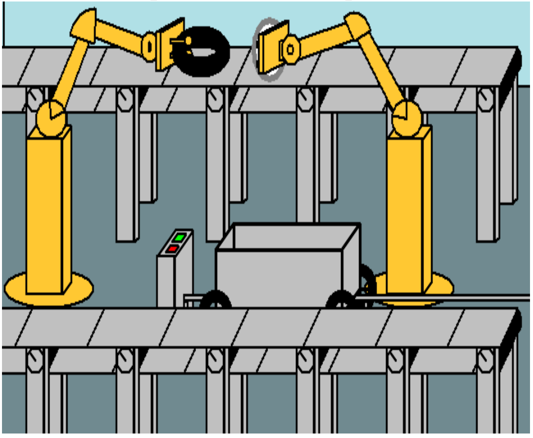

Fig. 2 Scene for cooperative mission

Scene 3: assembly line planning

In this example, shown in Fig. 3, the robot arms from scene 2 must assembly an ellipsoid object with the oval objects, on a conveyer belt. Both robots must be moved simultaneously to positioning the oval object in the gap of the ellipsoid object and to avoid the collision. The assembly line contains a support structure that is moving over the conveyer belt in the some direction to the assembled parts.

The moving structure may become an obstruction that causes the robots to reactively modify its path, to avoid the collision.

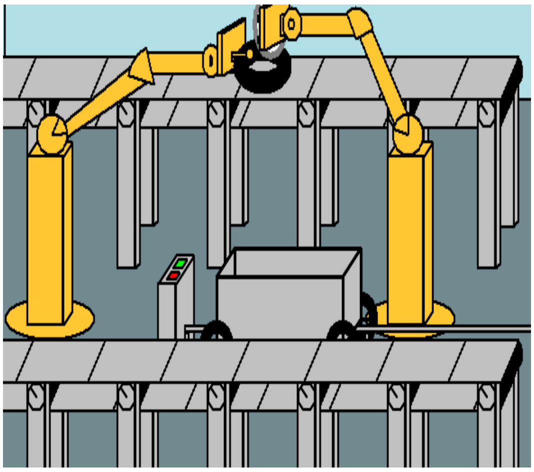

Fig. 3 Scene for assembly line planning

Since the usual models of the cooperative robots are rather complex and may have thousands of objects in the world, the algorithm as described in [6, 7], becomes an essential component to generate a realism of motions.

This image sequence shows discrete positions from our dynamic simulation application. In order to show the generality of our approach, we compute layouts of several kinds of geometric models. We used these layouts directly without any modification to the runtime application.

View-dependent rendering and simplification are frequently used for interactive display of massive models.

These algorithms pre-compute a multi-resolution hierarchy of a large model (e.g. a vertex hierarchy).

Figures 1, 2 and 3 show various instant blitz obtained during such virtual reality sessions, with only the inverse dynamics is applied, resulting in simulation.

6. CONCLUSIONS AND FUTURE WORKS

We propose an original idea that allows us to transfer the joint trajectory of each virtual robot to the corresponding joint of the real (physical) robot. With other worlds we prepare the free-trajectories for the pair of the virtual cooperative robots; these trajectories can be transferred to the pair of the physical cooperative robots.

In the real world, the programming pathway of the physical robots can be realized using the pathway of the virtual robot prototypes. Therefore, the virtual robot behavior must be specifically taken into account in order to guarantee that collisions are avoided. Each real robot may be able to communicate with her virtual corresponding entity and may imitate their intents [10, proposal patent].

In our collaborative work systems, the virtual robot prototypes are used mainly as an intermediate result for calculating the "nearest neighbors" and the potentially intersecting areas in an eventually collision of the virtual robots. We compute PCS of virtual objects that are either partly cover or are in close proximity. If an object does not belong to the PCS, it implies that this object does not collide with any object. As an alternative of testing each object pair in the PCS for exact partly cover, we more than use the visibility formulation to identify the potentially intersecting virtual areas among the objects.

Based on this property, we realized a programming platform for real (physical) robots based on collaborative virtual robots pair, that need to be checked for exact collision detection. This platform needs to compute the exact overlapping area of the virtual cooperative robots, as collision response.

BIBLIOGRAPHY

[1] **C. Fulgenzi, A. Spalanzani, C. Laugier,** "Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid". In Proc. IEEE Int. Conf. on Robotics and Automation, pp. 1610–1616, 2007.

[2] **N. K. Govindaraju, S. Redon, M. C. Lin and D. Manocha**, "CULLIDE: Interactive Collision Detection Between Complex Models in Large Environments using Graphics Hardware", M. Doggett, W. Heidrich, W. Mark, A. Schilling (Editors), Graphics Hardware, 2003.

[3] **D. Hennes, D. Claes, W. Meeussen, K. Tuyls**, Multi-robot collision avoidance with localization uncertainty, In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4– 8, 2012, Valencia, Spain.

[4] **Y. Abe, M.Yoshiki**, "Collision avoidance method for multiple autonomous mobile agents by implicit cooperation". IEEE/ RSJ, Int. Conf. Intelligent Robots and Systems, pp.1207-1212, 2001.

[5] **K. O. Arras, J. Persson, N. Tomatis, R. Siegwart**, " Real-Time Obstacle Avoidance for Polygonal Robots With a Reduced Dynamic Window" in Proc. Int. Conf. on Robotics and Automation, Washington DC, May 2002,.

[6] **N. Galoppo,** "Animation, Simulation, and Control of Soft Characters using Layered Representations and Simplified Physics-based Methods" Dissertation submitted to the faculty of the University of North Carolina Chapel Hill, 2008.

[7] **S.E. Yoon**, "Interactive Visualization and Collision Detection using Dynamic Simplification and Cache-Coherent Layouts". Dissertation submitted to the faculty of the University of North Carolina, Chapel Hill, 2006

[8] **J. van den Berg et al.**: Reciprocal n-body collision avoidance, Robotics Research, The 14th International Symposium ISRR, Springer Tracts in Advanced Robotics, vol. 70, Springer-Verlag, May 2011, pp. 3-19.

[9] **J. Snape, J. van den Berg, S. J. Guy, and D. Manocha**, "Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles", IEEE/RSJ Int. Conf. Intelligent Robots and Systems, St. Louis, Mo., 2009.

[10] **A. Fratu, M. Fratu**.: "Method and installation for joints trajectory planning of a physical robot arm" (proposal - patent) unpublished.

[11] **J. .Snape et al**. "Independent navigation of multiple robots and virtual agents. In Proc. of the 9th Int. Conf. on Autonomous Agents and Multi agents Systems (AAMAS 2010), Toronto, Canada, May 2010.

About the authors

Prof. Eng. **Aurel FRATU,** PhD.
Transilvania University of Brasov, 500036 Brasov, Romania
Automatics and Information Technology Department
email: fratu@unitbv.ro,

Main strengths of research activities: Modeling and simulation, Robots control systems, Digital control systems,

Prof. Eng. **Michel DAMBRINE,** PhD.
University of Valenciennes, F-59313 Valenciennes, France
Industrial and Human Automation, Mechanics and Computer Science (LAMIH) Laboratory
email: michel.dambrine@univ-valenciennes.fr

Main strengths of research activities: Stability of nonlinear systems, Robust control of uncertain nonlinear system, Real time robot control.