Motion planning for the large space manipulators with complicated dynamics*

Igor Belousov
HP Labs
52/1 Kosmodam. nab.
Moscow, Russia
belousov@keldysh.ru

KIAM-RAS
4, Miusskaya Sq.

Claudia Estevès and Jean-Paul Laumond
LAAS-CNRS
7, av. du Colonel Roche, 31077 Toulouse, France
{cesteves, jpl}@laas.fr

Etienne Ferré
Kineo C.A.M.
Prologue - La Pyreneenne, BP27201, 31672 Labège, France
eferre@kineocam.fr

***Abstract* – This paper deals with motion planning algorithms for the large space robot manipulators with complicated dynamic behavior. We propose two "two-stage" iterative algorithms, which provide collision-free robot motion taking into account robot's dynamics. The approach is based on new efficient methods for robot manipulator dynamics simulation and probabilistic methods for motion planning in highly cluttered environments. The algorithms are applicable for the robot manipulators of general class with arbitrary kinematics and dynamics parameters. We have demonstrated the approach for a particular task of servicing the satellite by a large space manipulator. This task is one of the most challenging since large space manipulators have extremely complicated dynamic behavior caused by elasticity of their structure, huge payloads they work with and zero-gravity conditions. Experiments involving a 15.5 meters long manipulator carrying a satellite inside a space shuttle with clearance less than 3 cm are presented. Several movies will demonstrate the results.**

Index Terms – Motion planning. Space robots. Robot dynamics.

I. INTRODUCTION

Robot manipulators are important for performing various servicing operations in space. They allow to minimise dangerous and fatigue extra vehicular activity (EVA) of astronauts and to perform some operations that would be impossible without such kind of robots [1]. Space Shuttle's Remote Manipulator System (RMS) is in operation for many years. Huge Canadian manipulator Canadarm2 is already used for assembling and servicing of the International Space Station (ISS). Japanese manipulator JEMRMS and European manipulator ERA should be launched to the ISS in 2006-2007 to perform servicing operations at the Japanese, Russian and European segments of the ISS. Preliminary laboratory-based development and testing is essential for the tasks to be performed by these robots in orbit.

The Virtual Robotic Test-bed (VRT), which provides a possibility to support the solution of this problem, has been built several years ago in the Robotics Laboratory of Keldysh Institute of Applied Mathematics (KIAM) [2], in accordance with project "Servicing" of Russian Space Agency. The goal of its development was training the astronauts to control on-board manipulator of the Russian space shuttle Buran. The VRT allows to compute the motion of the space manipulator *grip* and to execute it in real-time using the grip of the industrial robot for physical imitation of the computed motion (accounting for zero-gravity). It is possible to use the VRT for any type of the large space manipulators.

Some disadvantage of the VRT was in its possibility to control the robot only in *teleoperation* mode and lack of *automatic motion planning in the presence of obstacles*. The latter task is extremely complicated because of complicated dynamic behaviour of the large space manipulators due to elasticity of their structure, huge payloads they work with and zero-gravity conditions. Solving this task is of great importance for any space manipulators since they operate in cluttered working areas. The motivation for this research and present article is to solve this problem, summarizing authors' achievements on mathematical modelling and physical simulation of the large space manipulators dynamics, and in motion planning for robotic systems. The goal is to combine motion planning algorithms with dynamical models of manipulators to provide collision-free robot motion taking into account robot's dynamics.

Collision-free path planning for a robot manipulator in highly cluttered environment is already a challenging task, even at the kinematics level. When considering the dynamics of the robot – the task of collision free motion generation becomes much more complicated. This problem refers to a *kinodynamic motion planning* [3]. Because of tremendous complexity of this problem it still remains open for real-size applications, though some progress has been already achieved.

Kinodynamic motion planning could be implemented by one of the following approaches – "*two-stage planning*" and the "*state-space formulation"* [4]. In the "two-stage planning", an initial path is calculated to satisfy kinematic constraints, and then, at the second stage, optimisation of this path is performed to satisfy dynamic constraints and provide collision-free dynamic trajectory [5-7]. In the "state-space" algorithms final trajectory is calculated taking into account dynamic constraints from the very beginning [8-10]. A subclass of state-space algorithms deals with incomplete knowing of the robot environment, which is refined on-line using different sensors [11]. One of the most promising techniques for the state-space formulation is using bi-directional RRTs (Rapidly-Exploring Random Trees). This

* This work is partially supported by the IST-2001-39250 MOVIE project from the EC, by CNRS grant to I.Belousov and Mexican Conacyt grant to C.Estevès.

approach provides reasonable calculation cost for the objects with relatively high number of DOF [12, 13], though generated trajectories could be not optimal. We will mention here also several motion planning algorithms specially developed for the space robot applications, such as NASDA's ETS-VII experiment [1, 14], and new German project TECSAS [15].

Methods for space robot motion planning with dynamics, which will be presented in this article, belong to the first approach, "two-stage planning". The methods are applicable for the robot manipulators of general class with arbitrary kinematics and dynamics parameters. We have demonstrated the approach for a particular task of servicing the satellites by a large space manipulator. Two tasks have been considered: posing the satellite to the orbit by means of Buran's on-board manipulator, and it's docking inside the cargo bay of the Buran space shuttle. Firstly, we calculate collision-free paths to solve both tasks at the kinematics level. Then, considering these paths like the control inputs, we simulate real dynamic motion of the robot manipulator with the satellite in the end-effector, and check for possible collisions. Two methods for avoiding collisions have been proposed – tuning the bounds of control points located on the kinematic path ("*base points*" control) and "*running point*" control. Our methods are based on the fast and general algorithms for simulation of the robot manipulator dynamics [16] and randomised motion planning [17]. The contribution of the paper is as follows:

- To propose an original methods for robot dynamics simulation
- To propose an integrated approach accounting for collision avoidance as well as for dynamic constraints
- To solve a real-size problem dealing with highly cluttered environment.

The paper is organised as follows. Second section is devoted to the description of the new efficient methods for robot manipulator simulation. In particular, new methods for Lagrangian formulation of the robot dynamics and for fast integration of the robot dynamics equations are presented. These algorithms have general form and could be applied for any open chains of articulated bodies. Motion planning algorithms, which allow motion planning for robot manipulators in highly cluttered environments are presented in the third section. Motion planning is based on a general efficient probabilistic diffusion algorithm working in the configuration space of the considered system [17]. Forth section presents the algorithms for generation collision-free trajectories for manipulators with complicated dynamical behaviour. Experimental results on real-size system are presented.

II. The Methods of Real-Time Simulation of the Space Robot Dynamics

Large weight and size of RMS, Canadarm2, JEMRMS, ERA and Buran on-board manipulator, large weight of the objects (up to 100 tons), to be carried by them, elasticity of the links and mechanical gears, can lead to great deviation of the real motion from the desired one. These dynamic features as well as zero-gravity conditions have to be simulated when planning the motion of space manipulators. The dynamic simulation we have described comprises an efficient formulation of dynamic equations and fast integration algorithms. These algorithms could be applied for various types of manipulators and open-loop chains of the articulated bodies.

A. Formulation of the Robot Manipulator Dynamics

During last 30-40 years a lot of algorithms for robot dynamics calculation have been proposed. The comprehensive review of the developed methods is given by Featherstone and Orin in [18]. Our algorithm provides high computational efficiency, allows obtaining the dynamics equations for any type of the open-chain manipulators with parallel or perpendicular neighbor joints (both rotational and sliding). We used Lagrange 2nd order equations to get closed-form formulation, suitable for solving both direct and inverse dynamics tasks. The particularity of our approach is in using 3x3 rotational matrices instead of commonly used 4x4 homogeneous matrices.

The dynamic equation could be presented at the following form:

$$\sum_{s=1}^{n} d_{ks}\,\ddot{q}_s + \sum_{s,t=1}^{n} h_{kst}\,\dot{q}_s\,\dot{q}_t + p_k = \tau_k$$

or, in the matrix representation: $D(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{p}(\mathbf{q}) = \boldsymbol{\tau}$

Here D($\mathbf{q}$) is the symmetric, positively defined inertia matrix of the manipulator with the components:

$$d_{ks} = \sum_{i=\max(k,s)}^{n} \left[m_i \left(\sum_{p=k}^{i} V_{pk}\mathbf{r}_p, \sum_{l=s}^{i} V_{ls}\mathbf{r}_l \right) + Tr(V_{ik}\,\tilde{J}_i\,V_{is}^T) \right]$$

$\mathbf{h}(\mathbf{q},\dot{\mathbf{q}})$ is the vector of the Coriolis and centrifugal forces:

$$h_k = \sum_{s,t=1}^{n} h_{kst}\,\dot{q}_s\,\dot{q}_t \ ,$$

$$h_{kst} = \sum_{i=\max(k,s,t)}^{n} m_i \left(\sum_{p=k}^{i} V_{pk}\mathbf{r}_p, \sum_{l=\max(s,t)}^{i} V_{lst}\mathbf{r}_l \right) + Tr(V_{ik}\,\tilde{J}_i\,V_{ist}^T)$$

$\mathbf{p}$ is the vector of gravitational forces with the components:

$$p_k = \sum_{i=k}^{n} -m_i \left(\mathbf{g}, \sum_{p=k}^{i} V_{pk}\mathbf{r}_p \right)$$

In the above formulas:

$V_{ps} = C_1\,C_2 \ldots C_{s-1}\,Q_s\,C_s \ldots C_p$

$$Q_i = \left\{ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \right\}$$

$C_i = \{C_x, C_y, C_z\}$ – rotation matrices 3x3

$$V_{pks} = \frac{\partial}{\partial q_s} V_{pk}$$

$$\mathbf{r}_p = \begin{cases} \mathbf{l}_p \text{ while } p < i \\ \mathbf{r}_{C,i} \text{ if } p = i \end{cases} \qquad (i=1,..,n)$$

$$\tilde{J}_i = J_i - m_i \mathbf{r}_{C,i} \mathbf{r}_{C,i}^T$$

$\mathbf{r}_{C,i}$: radius-vector of the center of mass of the link *i* in the link's Reference Frame (RF);
$\mathbf{l}_i$: vectors of translation between neighbor joints of the manipulator;
m_i: mass of the link *i* of the manipulator;
J_i: inertia matrix of the link *i* of the manipulator.

B. Integration Algorithms

Main difficulties in the dynamic simulation of the large space manipulators are connected with the numerical integration of dynamic equations because of their essential stiffness. Stiffness is caused by a large size of manipulator links, which leads to essential differences in the actuator torques. Usually explicit methods (such as Runge-Kutta 4th-order method) are used for integration of space robot dynamic equations. To provide stability of a calculation scheme in that case, a small integration step (about $10^{-3} - 10^{-5}$ sec) should be chosen.

The implicit integration methods permit to increase integration step providing calculation stability and sufficient accuracy. We have used Euler and Adams (second order) implicit methods for the integration of the dynamic equations of space manipulators [16]. As opposed to the methods we've already presented in [2] and [16], currently we've extended them to cope not only with velocity-based controller (used for teleoperation mode), but also with PD controller (to allow simulation while controlling the robot in automatic regime).

The dynamic model takes into account elasticity of joints and nonlinear elements in actuators and mechanical gears - friction, backlashes, limits on maximum values of torque in joints, current and voltage in drives. This causes appearance of the domains for generalized coordinates $\mathbf{q}$ and $\mathbf{w} = \dot{\mathbf{q}}$.

Crossing the boundaries of the domains leads to the changing of the structure of the right-hand sides of the dynamic equations. To provide correctness of integration, special algorithms for determination of switch points were developed. Calculation of these points was implemented by means of interpolation algorithms. The method dealing with voltage limits violation has been modified. For PD controller these limits depend on 2 variables q_i and ω_i as opposed to 1 variable ω_i for velocity-based controller. Currently we calculate the switch points as intersection of the integral curve with the plane $\left|u_i^{\max}\right| = \left|\alpha_i\ (\omega_i^{pr} - \omega_i^{dr}) + \beta_i\ (q_i^{pr} - q_i^{dr})\right|$.

Dynamic equations of the space manipulator with elastic joints are:

$$\begin{cases} \mathrm{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q},\dot{\mathbf{q}}) = E_c \boldsymbol{\Delta} \\ \mathrm{E}_{J^{\mathrm{dr}}} \dot{\mathbf{w}}^{dr} = \mathbf{M}^{dr} - E_c \boldsymbol{\Delta} \end{cases} \qquad (1)$$

where $\mathrm{D}(\mathbf{q})$ is the manipulator inertia matrix, $\mathbf{H}(\mathbf{q},\dot{\mathbf{q}})$ is the vector of Coriolis and centrifugal forces, $\boldsymbol{\Delta} = (\mathbf{q}^{dr} - \mathbf{q})$ is the deformation vector, $\mathbf{q}^{dr}$ defines position of D.C. rotors, E_c is the matrix of joints stiffness, $\mathbf{M}^{dr}$ is the torques in the drives and $E_{J^{dr}}$ is the diagonal inertia matrix of D.C. rotors.

Using Euler implicit method, we obtain:

$$\begin{cases} \mathbf{w}^{(n+1)} = \mathbf{w}^{(n)} + h_e \left[D^{(n)}\right]^{-1} (E_c \boldsymbol{\Delta}^{(n+1)} - \mathbf{H}^{(n)}) \\ \mathbf{w}^{dr(n+1)} = \mathbf{w}^{dr(n)} + h_e E_{1/J^{dr}} (\mathbf{M}^{dr(n+1)} - E_c \boldsymbol{\Delta}^{(n+1)}) \end{cases} \qquad (2)$$

$$\begin{cases} \mathbf{q}^{(n+1)} = \mathbf{q}^{(n)} + h_e \mathbf{w}^{(n+1)} \\ \mathbf{q}^{dr(n+1)} = \mathbf{q}^{dr(n)} + h_e \mathbf{w}^{dr(n+1)} \end{cases} \qquad (3)$$

h_e is an integration step; $D^{(n)} = D(\mathbf{q}^{(n)}); \mathbf{H}^{(n)} = \mathbf{H}(\mathbf{q}^{(n)}, \dot{\mathbf{q}}^{(n)})$.

Note: obviously, we can consider $D^{(n)} = D^{(n+1)}$ and $\mathbf{H}^{(n)} = \mathbf{H}^{(n+1)}$ because of their small changes during integration step.

Since $\boldsymbol{\Delta}^{(n+1)} = \mathbf{q}^{dr(n+1)} - \mathbf{q}^{(n+1)} = \boldsymbol{\Delta}^{(n)} + h_e(\mathbf{w}^{dr(n+1)} - \mathbf{w}^{(n+1)})$, equations (2) depend only on $\mathbf{w}^{dr(n+1)}$ and $\mathbf{w}^{(n+1)}$. Solving this system, we get:

$$w_i^{dr(n+1)} = \mathrm{s}_\mathrm{i}^{\mathrm{w}} w_i^{(n+1)} + \mathrm{s}_\mathrm{i}^{\mathrm{c}}, \mathrm{i} = 1,.., \mathrm{n} \qquad (4)$$

While calculating constants $\mathrm{s}_\mathrm{i}^{\mathrm{w}}, \mathrm{s}_\mathrm{i}^{\mathrm{c}}$ we must take into account possible violation of limits for voltage and current.

To get the solution of (2), we substitute (4) in the first equation of system (2) and have the equation:

$$\tilde{D}\mathbf{w}^{(n+1)} = D^{(n)}\mathbf{w}^{(n)} + \mathbf{b} \qquad (5)$$

where $\tilde{d}_{ij} = d_{ij}^{(n)}$ (for $\mathrm{i} \neq \mathrm{j}$);

$$\tilde{d}_{ii} = d_{ii}^{(n)} - h_e^2 \mathrm{c}_\mathrm{i}(\mathrm{s}_\mathrm{i}^{\mathrm{w}} - 1);$$

$$b_i = h_e \mathrm{c}_\mathrm{i}(q_i^{dr(n)} - q_i^{(n)} + h_e \mathrm{s}_\mathrm{i}^{\mathrm{c}}) - h_e H_i^{(n)}$$

In such a way we reduce solving of 4th-order system (1) to solving of 1st -order equation (5).

Using similar implicit algorithms for integration of dynamic equations of space manipulator with the above nonlinearities and joint elasticity, we can increase integration step up to 100 times (from $h_r = 10^{-3} - 10^{-5}$ sec for explicit Runge-Kutta 4th-order method with accuracy control, to h_e =0.05-0.15 sec for the above implicit method). Such increase of integration step doesn't affect calculation stability and gives the following errors for the particular case of Buran's manipulator simulation:

$\delta_p < 0.7$ mm, $\delta < 1\%$, $\delta_o < 0.1$ deg

where:

$\delta_p = \underset{t<T}{Max}\{\delta_p(t)\}$ is the absolute position error

$\delta = \delta_p / D$ is the relative position error

$\delta_o = \underset{t<T}{Max}\{\delta_o(t)\}$ is the absolute orientation error

Here T is the simulation time, δ_p *(t)* and δ_o *(t)* are the differences in grip position and orientation, calculated by explicit and implicit methods for the time instant *t*, and *D* is the diameter of the simulation working area.

III. Motion planning

The path planning algorithm we use is dedicated to highly constrained spaces where the motion to be computed is close to the contact space [17]. The algorithm is iterative. A first path is computed allowing some penetration in the obstacles. Then the current paths are iteratively re-shaped by decreasing the allowed penetration threshold. The cases of failure of the iterative process are automatically detected and solved.

The approach benefits from several principles:

- As collision checking is concerned, a critical problem is to perform efficient collision checking not only for configurations (see overviews in [19, 20]) but also for local paths. Exact collision checking along computed paths has been recently addressed in [21]: path collision checking is performed with a static collision checker while the (usually costly) iterative process is speeded up thanks to distance computations. We have extended the approach to account for the user-defined imposed clearance constraints.
- To overcome the expansive cost of configuration and path collision checking, some approaches have been defined to put back the tests and then to avoid useless computations. This is the case of the lazy approaches where the algorithms put back collision checking as long as the probability of failure is high [22, 23]. Our approach consists in starting from a rough solution path and iteratively refining it. The iterative procedure is based on an original penetration distance control. When the refinement procedure fails (i.e. when the current path cannot be locally re-shaped into a collision-free one), then the search re-starts with a roadmap composed of the portions of the path that are collision-free.[†]
- Another key point is the control of the diffusion process: how to steer the diffusion process without introducing useless side effects? For instance, defining a new diffusion direction at random by fixing a new configuration goal (as in [25]) gives rise to a bias in introducing implicit bounding boxes on the translation parameters. The solution in [26] depends on a local grid whose resolution appears as a parameter to be tuned. The solution we propose is parameter free (see [17] for details).
- Finally our refinement procedure for path reshaping follows the same idea as the variation approach (introduced in [27]) where the search is performed by iteratively growing formerly shrunk obstacles. In our approach, the growing process is automatically controlled. Moreover the failures due to the closure of passages at some stage of the growing are automatically solved.

The algorithm is general. It works for free-flying objects as well as for articulated mechanisms. The output of the algorithm is a finite sequence of *via points*, connected by straight lines segments in the configuration space. We define the image of a via point in the 6-dimensional space of the end effector as being a *base point*.

The figures below demonstrate the output of our software package, where the developed methods for motion planning have been realized. The calculated collision-free path to dock the satellite by the Buran on-board manipulator inside the Buran cargo bay appears in Figures 1 and 2.

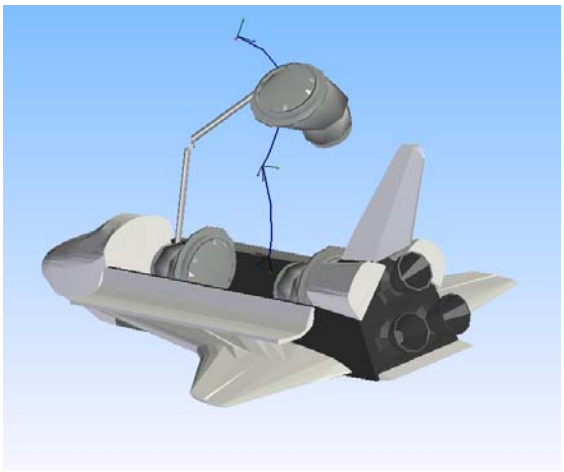

Fig. 1 Collision-free path with the so-called *"base points"* (without accounting for system dynamics).

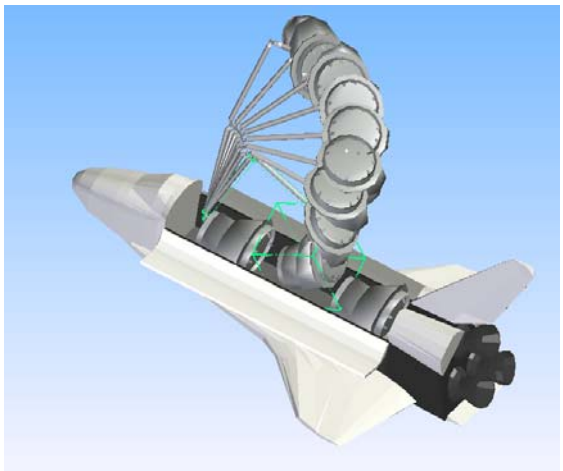

Fig. 2 Swept volume along the collision-free path.

[†] That kind of procedure has been recently introduced in [24] to improve the connectivity of roadmaps.

Figure 3 illustrates the collision, detected by our collision-checker after a direct application of the dynamics. How to control collision avoidance within the dynamic simulation? This question is the purpose of the next section.

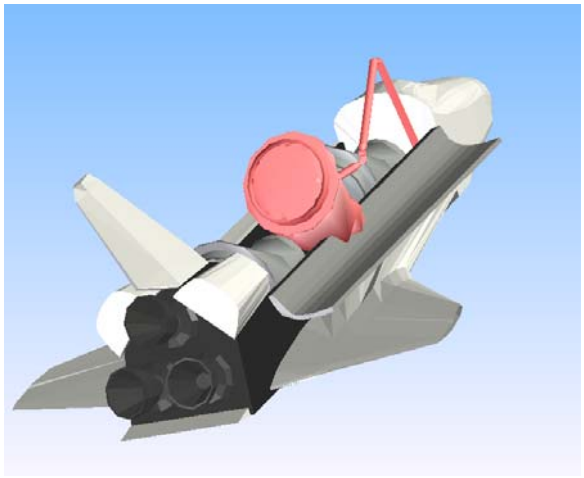

Fig. 3 Detection of the collision for real dynamic trajectory.

IV. Dynamic Approximation of the Kinematic Path

This section is devoted to the description of two methods for motion planning with dynamical constraints. These methods provide collision-free robot motion in the environment with obstacles taking into account robot's dynamics. Methods are based on the algorithms for dynamic simulation and motion planning, described at the previous sections. A "two-stage planning" approach for solving the problem has been chosen. It means that at the first stage an initial path is calculated to satisfy only kinematic constraints (collision-free path is calculated). Then, at the second stage, approximation of this path is performed to satisfy dynamic constraints while preserving collision avoidance.

Dynamic behaviour of the Buran on-board manipulator, used in our experiments, is extremely complicated due to elasticity of its joints, large mass and inertia of its links and presence of non-linear elements in mechanical gears (see section 2). In particular, the following dynamic features have been revealed during investigation of the dynamic model of the Buran manipulator and experiments with real robot on the air-bearing test-bed:

- Oscillation with large amplitude (up to 50 cm) and small frequency (0.2-1 Hz)
- Large accelerating and decelerating path, resulting from the important inertia of the manipulator links.

These dynamical features significantly complicate the problem of kinodynamic motion planning for the robot. Two methods to solve the problem are presented below.

A. *Using "Base points": an interactive approach.*

The idea of this method is the following. We consider the sequence of the "via points" (in the configuration space), corresponding to the path computed at the first stage. Each *via point* in the configuration space corresponds to the so-called *"base point"* in the workspace of the end-effector. Base points are 6-dimensional (3 parameters for position and 3 parameters for orientation). The input for the dynamics simulator is then the corresponding sequence of the base points $\boldsymbol{B_i}$. In the example of Figure 1 (section 3) there are 3 base points. Besides that, for each point $\boldsymbol{B_i}$ we define:

- Proximity spheres for position R_i and orientation of the robot grip. If the grip is inside these spheres it is considered that current base point $\boldsymbol{B_i}$ is achieved and robot controller will take next point $\boldsymbol{B_{i+1}}$ as the goal.
- Extended proximity sphere for position $R_i^+ > R_i$.
- Trajectory speed V_i on the segment ($\boldsymbol{B_{i-1}}$, $\boldsymbol{B_i}$).
- Desired time t_i to move the robot from $\boldsymbol{B_{i-1}}$ to $\boldsymbol{B_i}$.

Starting from the initial position $\boldsymbol{B_0}$ robot grip is "attracted" to the sphere R_1^+ of the current base point $\boldsymbol{B_1}$. When arriving inside the sphere R_1^+, control algorithm decreases the coefficients α_j and β_j of PD-controller for the joint speed and position in order to provide accurate robot motion to the sphere R_1 with the suitable errors for the grip position and orientation. When arriving inside the sphere R_1, control algorithm switches the control point to the next one, i.e. $\boldsymbol{B_2}$, and so on. Square in Figure 4 represents switching point for the controller.

Control input is calculated in the following form:

$$u_j = \alpha_j(\omega_j^{pr} - \omega_j) + \beta_j(q_j^{pr} - q_j) \qquad (6)$$

where q_j, ω_j are angle and speed for the joint j, q_j^{pr}, ω_j^{pr} - program values of angle and speed for the joint j. Parameters q_j^{pr} directly defined by the current base point. Parameters ω_j^{pr} are defined using desired trajectory speed (solving the inverse kinematic problem for velocities).

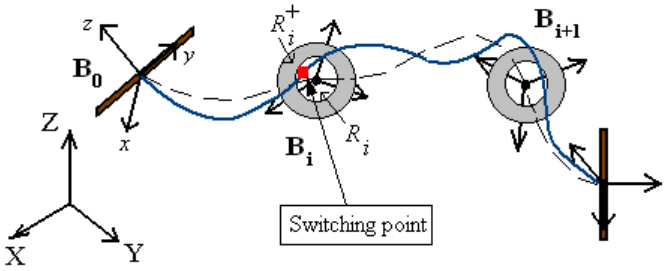

Fig. 4 Kinematic path (dotted line) and dynamic trajectory (solid line).

Figure 4 gives the idea of the algorithm. Inside gray regions decreased values of coefficients α_i and β_i are used.

The interactive procedure is used to choose appropriate parameters (t_i, V_i, R_i, R_i^+) to minimize overall time needed to achieve the goal position (with admissible error) keeping collision-free motion. While integrating the equation of robot dynamics we calculate for each integration step minimal distance between the robot and the closest obstacle d_i and check the condition $d_i > 0$ for each point on the path.

The main advantage of the methods is possibility for the user to manually adjust the above parameters. The method is currently under implementation.

B. "Running Point" Control: an automatic approach.

As opposed to the "base points" method described above, here we make discretization of the path, computed at the first stage, defining "input" set of control points for dynamic simulation (Figure 5).

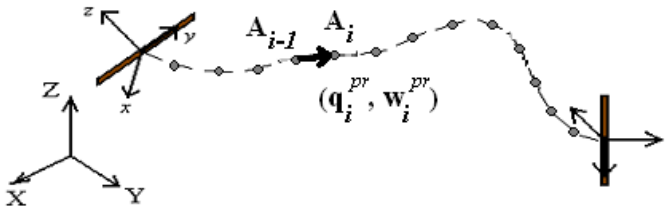

Fig. 5 "Running point" algorithm.

The number of control points depends on the desired maximal linear and angular velocities for the robot grip V^{max} and W^{max}. Let the robot control period be t_{contr} (defines the frequency of the control inputs change at the high level of the robot control system). Then the approximate number of needed control points is $N=[T/t_{contr}]$, where:

$T=\text{Max}(\text{Path\_Length}/V^{max},\text{Grip\_Orientation\_Change}/W^{max})$.

Parameters V^{max} and W^{max} define how fast the robot will follow the trajectory. Using these parameters we can define the profile for linear $\mathbf{V}_i$ and angular $\mathbf{W}_i$ velocities of the robot grip along the trajectory:

$$\mathbf{V}_i = d_i / d_i^{\max} V^{max} \mathbf{A}_{i-1}\mathbf{A}_i / ||\mathbf{A}_{i-1}\mathbf{A}_i||$$

$$\mathbf{W}_i = d_i / d_i^{\max} W^{max} \mathbf{n}$$

where $d_i^{\max} = Max(d_i)$, $\mathbf{n}$ is the vector of rotation between 2 grip orientations in points $\mathbf{A}_{i-1}$ and $\mathbf{A}_i$. Such a choice of $\mathbf{V}_i$ and $\mathbf{W}_i$ allows accelerating the robot when it is far enough from the obstacles (i.e. the coefficient $d_i / d_i^{\max}$ is about 1), and vice versa, the robot is decelerating when the clearance d_i is small (i.e. the coefficient $d_i / d_i^{\max}$ is correspondingly less than 1).

Then for each point on the trajectory we calculate desired values for joint velocities ω_j^{pr}, multiplying vector $(\mathbf{V}_i, \mathbf{W}_i)$ by the inverse Jacoby matrix of the manipulator. At this stage control law (6) is fully defined, since q_j^{pr} have been already calculated at the first stage. So, the approximation problem here is simply choosing admissible velocities to keep collision-free motion and to minimize overall time for trajectory following. We've used a *dichotomy* method to cope with that problem. This means that firstly we try maximum values of linear and angular velocities, which can be realized physically by the robot. If these values lead to collisions during dynamics simulation we divide them by 2, try new values, and so on. Lets note, that in this algorithm we don't demand the robot grip to arrive to the proximity of the current control point, like in the algorithm above – regardless the grip will achieve the control point or not, the control system switch it to the next one with fixed frequency $[1/t_{contr}]$.

The resulting dynamic trajectories for the task of docking the satellite inside the Buran cargo bay are presented in Figures 6 and 7. The number of control "running" points necessairy to guarantee collision avoidance is 26. The algorithm provides the dynamical solution for the extremely tight space – the range of *minimal distances* between the robot with satellite and workspace varied from just *2-3 cm* to *18 cm* for 15.5 meters long manipulator with 4 meters long satellite. Trajectory length is 18.2025 m, total time to perform the operation is 98.09 sec. The average grip linear speed is *18.55 cm/sec*. This value exceeds maximum speed for Buran manipulator with payload (10 cm/sec) and just a little below its maximum possible speed, equal to 30 cm/sec.

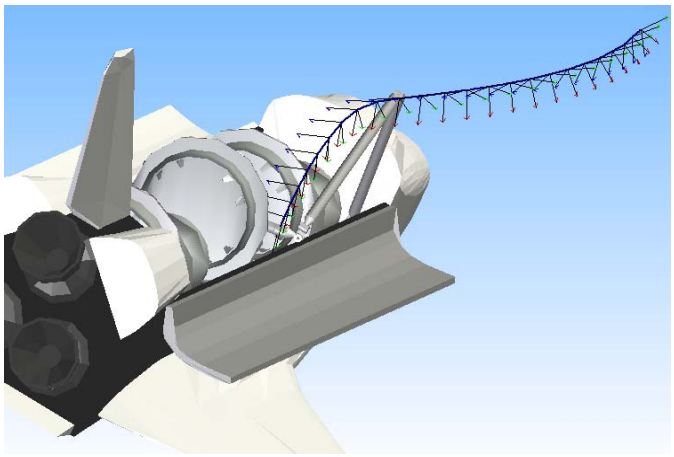

Fig. 6 The *running points* and the collision-free dynamic trajectory.

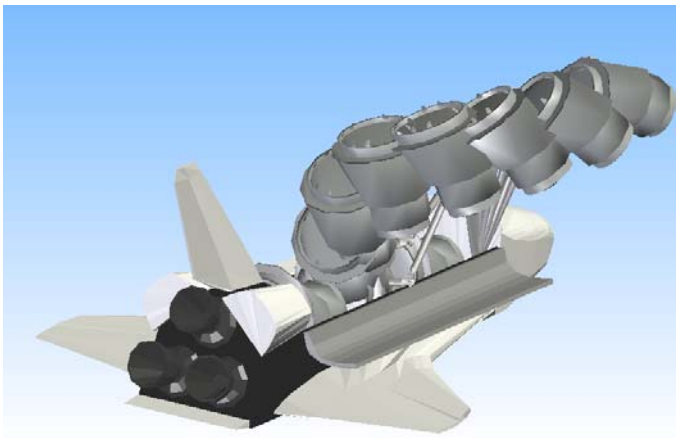

Fig. 7 Swept volume along the collision-free dynamic trajectory (collision, presented in Figure 3, disappeared after applying the *running point* algorithm).

This method allows fast *automatic* dynamic approximation of the kinematics path with collision avoidance. Though the linear speed of the robot end-effector can be less that one, provided by the "base points" algorithm, ultimately the operation (docking the satellite) is usually

accomplished faster, due to quasi-optimal choice of the speed in the "running point" algorithm.

V. Conclusion

The principal novelty of the results, presented in this article, is as follows:

- New efficient motion planning algorithms for robot manipulators with complicated dynamics have been developed
- Its consistency has been demonstrated for the most difficult case of servicing tasks, performed by a large space manipulator in highly cluttered environment with a clearance less than 3 cm.
- New efficient formulation of the robot manipulator dynamics and fast implicit algorithm for integration of the dynamics equations has been proposed.

Movies with the experimental results can be found at http://www.laas.fr/~jpl/dynamic2005.

Future research will be focused on developing the algorithms that will take into account robot manipulator dynamics at the motion planning stage, using state-space formulation and RRT technique. Special efforts will be made to provide *real-time* dynamic motion planning that will allow planning in the dynamic environments with moving obstacles.

Acknowledgment

The authors would like to thank the head of the Robotic Department of the Keldysh Institute of Applied Mathematics Academician Dmitry Okhotsimsky for important advices concerning space robots simulation. Part of the work has been done during the stay of Igor Belousov at LAAS-CNRS (Toulouse, France).

References

[1] G. Hirzinger, B. Brunner, R. Lampariello, K. Landzettel, J. Schott, B.-M. Steinmetz, "Advances in orbital robotics", *Proc. IEEE International Conference on Robotics and Automation ICRA'2000*, San Francisco, CA, April 2000, pp. 898-907.

[2] I. Belousov, V. Kartashev, D. Okhotsimsky, "Real time simulation of space robots on the virtual robotic test-bed", *Proc. 7th Intern. Conf. on Advanced Robotics ICAR'95*, Sant Feliu de Guixols, Spain, Sept. 20-22, 1995, pp. 195-200.

[3] B. Donald, P. Xavier, J. Canny, J. Reif, "Kinodynamic motion planning", *Journal of the ACM*, 40 (5), November 1993, pp. 1048-1066.

[4] J. Kuffner, "Motion planning with dynamics", *Physiqual*, March 1998.

[5] G. Sahar, J. Hollerbach, "Planning of minimum-time trajectories for robot arms", A.I. Memo No. 804, MIT Press, November 1984.

[6] J. Bobrow, S. Dubowsky, J. Gibson, "Time-optimal control of robotic manipulators along specified paths", *Int. Journal of Robotics Research*, Vol. 4, No. 3, 1985, pp. 3-17.

[7] Z. Shiller, S. Dubowsky, "On computing the global time-optimal motions of robotic manipulators in the presence of obstacles", *IEEE Trans. on Robotics and Automation*, Vol. 7, No. 6, December 1991, pp. 785-797.

[8] B. Donald, P. Xavier, "Probably good approximation algorithms for optimal kinodynamic planning for Cartesian robots and open chain manipulators", *Algorithmica*, 14(6), 1995, pp. 480-530.

[9] E. Frazzoli, M. Dahlen, E. Feron, "Real-time motion planning for agile autonomous vehicles", *AIAA paper*, 2000-4056.

[10] R. Kindel, D. Hsu, J.-C. Latombe, S. Rock, "Kinodynamic motion planning with moving obstacles", *Proc. IEEE Int. Conf. on Robotics and Automation ICRA'2000*, San Francisco, CA, April 2000.

[11] V. Lumelsky, A. Shkel, "Incorporating body dynamics into the sensor-based motion planning paradigm", *Proc. IEEE Int. Conf. on Robotics and Automation ICRA'1995*, Nagoya, Japan, May 1995, pp. 1637-1642.

[12] S. La Valle, J. Kuffner, "Randomized kinodynamic planning", *Int. Journal of Robotics Research*, 20 (5), May 2001, pp. 378-400.

[13] F. Lamiraux, E. Ferré, E. Vallee, "Kinodynamic motion planning: connecting exploration trees using trajectory optimization methods", *Proc. International Conference on Robotics and Automation ICRA'2004*, New Orleans (USA), April 2004, pp. .3987-3992.

[14] M. Oda, "Experiences and lessons learned from the ETS-VII robot satellite", *Proc. IEEE International Conference on Robotics and Automation ICRA'2000*, San Francisco, CA, April 2000.

[15] F. Cusumano, R. Lampariello, G. Hirzinger, "Development of tele-operation control for a free-floating robot during the grasping of a tumbling target", *International Conference on Intelligent Manipulation and Grasping*, Genoa, Italy, July 2004.

[16] I. Belousov, "Methods for robot manipulators simulation and control", Russian Academy of Sciences Journal of Differential Equations, Vol. 39, No. 8, 2003, pp. 1144-1145.

[17] E. Ferré, J.-P. Laumond, "An iterative diffusion algorithm for part disassembly", *Proc. International Conference on Robotics and Automation ICRA'2004*, New Orleans (USA), April 2004.

[18] R. Featherstone, D. Orin, "Robot dynamics: equations and algorithms", *Proc. IEEE Intern. Conf. on Robotics and Automation*, San Francisco, CA, April, 2000.

[19] M. Lin, D. Manocha, J. Cohen, S. Gottschalk, "Collision detection: algorithms and applications", *Proc. of the Intern. Workshop on Algorithmic foundations of Robotics WAFR'96*, 1996.

[20] P. Jimenez, F. Thomas, C. Torras, "3d collision detection: a survey", *Computers and Graphics*, vol. 25, No. 2, 2001.

[21] F. Schwarzer, M. Saha, "Exact collision checking of robots paths", *Proc. of the Intern. Workshop on Algorithmic foundations of Robotics WAFR'02*, 2002.

[22] R. Bohlin, L. Kavraki, "Path planning using lazy PRM", *Proc. IEEE International Conference on Robotics and Automation ICRA'2000*, San Francisco, CA, April 2000.

[23] G. Sanchez, J. Latombe, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking", *International Symposium on Robotics Research ISRR'01*, 2001.

[24] M. Akinc, K. Bekris, C. Chen, A. Ladd, E. Plakue, L. Kavraki, "Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps", *International Symposium on Robotics Research ISRR'03*, 2003.

[25] S. La Valle, J. Kuffner, "Rapidly-exploring random trees: progress and prospects", *Proc. of the Intern. Workshop on Algorithmic foundations of Robotics WAFR'00*, 2000.

[26] D. Hsu, J.-C. Latombe, R. Motwani, "Path planning in expansive configuration spaces", International Journal of Computational Geometry and Applications, vol. 9, No. 4/5, 1999.

[27] J. Barraquand, J.-C. Latombe, "A penalty function method for constrained motion planning", *Proc. IEEE International Conference on Robotics and Automation ICRA'94*, 1994.