Robotica (2010) volume 28, pp. 885–892. © Cambridge University Press 2009
doi:10.1017/S0263574709990737

Task-priority motion planning of underactuated systems: an endogenous configuration space approach

Adam Ratajczak*, Joanna Karpińska and Krzysztof Tchoń

Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50–372 Wrocław, Poland. E-mails: joanna.karpinska@pwr.wroc.pl, krzysztof.tchon@pwr.wroc.pl

(Received in Final Form: November 10, 2009. First published online: December 11, 2009)

SUMMARY
This paper presents a task-priority motion planning algorithm for underactuated robotic systems. The motion planning algorithm combines two features: the idea of the task-priority control of redundant manipulators and the endogenous configuration space approach. This combination results in the algorithm which solves the primary motion planning task simultaneously with one or more secondary tasks ordered in accordance with decreasing priorities. The performance of the task-priority motion planning algorithm has been illustrated with computer simulations of the motion planning problem for a container ship.

KEYWORDS: Underactuated system; Motion planning; Task priority; Endogenous configuration approach; Container ship.

1. Introduction
An underactuated robotic system is a robotic system which has more degrees of freedom than control inputs. In other words, the dimension of the state space of the underactuated system is greater than the dimension of its control space. For this reason, the motion planning problem calls for more sophisticated tools than in the case of fully actuated robotic systems.

The underactuated systems can be divided into two groups. The systems belonging to the first group are subjected to nonholonomic kinematic constraints of order 1 that assume the Pfaffian form. The mathematical model of such a system is represented by a driftless control system. Representative members of this group are wheeled mobile robots. The second group is characterized by the nonholonomic constraints of order 2 that arise from the system's dynamics. The mathematical model of a system from this group involves an affine control system with a nontrivial drift term. This group includes robotic manipulators with passive joints or with flexible arms, and also the mobile robots like surface and underwater robots, aircraft robots, hovercrafts, etc. The members of both groups are nonholonomic due to the presence of either kinematic or dynamic nonholonomic constraints, and are underactuated, since they have less control inputs than degrees of freedom. A general characterization of underactuated systems can be found in ref [1], the control problems involving underactuated manipulators have been addressed in ref. [2], a derivation of control algorithms for underactuated ships has been done in refs. [3, 4].

This paper introduces a task-priority motion planning algorithm for underactuated robotic systems. Its derivation relies on the endogenous configuration space approach.[5,6] The presented algorithm allows to plan the motion along with solving one or more additional tasks. These additional tasks may refer to ensuring a desirable quality of motion, keeping the configuration variables within some limits, avoiding singularities, preventing collisions, etc. Originally, the concept of prioritizing the subtasks for redundant manipulators was set forth in ref. [7], however, for the algorithm derivation a slightly different approach, coming from ref. [8], will be followed. The first attempt to control the underactuated systems using the endogenous configuration space approach has been made in ref. [9], and a motion planning problem of the underactuated ship with this approach has been addressed in ref. [10]. In this paper a complete derivation of the task-priority motion planning algorithm for underactuated systems is made. The obtained algorithm is then tailored to a motion planning problem of a container ship. The algorithm is tested with computer simulations involving a model of a container ship that comes from ref. [4]. The performance of task-priority-based control algorithms for the underwater robots is studied in refs. [11, 12].

Compared to the existing literature,[1–4] the benefits of the new algorithm are threefold. First, within the class of affine control systems the algorithm is not model specific. Second, the algorithm is based on the widely accepted technique of linear approximation. Third, its derivation proceeds along a well-established route of task-priority motion planning algorithms for holonomic manipulators.

The composition of this paper is the following. Section 2 deals with the modeling of underactuated systems. The motion planning problem is stated in Section 3. Main result, the derivation of a task-priority motion planning algorithm, is included in Section 4. An application to the motion planning of a container ship is presented in Section 5. The paper concludes with Section 6.

* Corresponding author. E-mail: adam.ratajczak@pwr.wroc.pl

2. Modeling of Underactuated System

Let a general dynamics equation of a robotic system be given

$$M(q)\ddot{q} + F(q,\dot{q}) = B(q)u, \tag{1}$$

where $q \in I\!R^n$ is a vector of internal coordinates, $u \in I\!R^m$ is the vector of control forces and torques, $M(q)$ denotes the inertia matrix, $F(q,\dot{q})$ collects centrifugal, Coriolis, friction, gravitational, buoyancy, hydrostatic, hydrodynamic, and other terms, $B(q)$ is a control matrix. Very often the dynamics equation is augmented with an output function

$$y = h(q,\dot{q}), \tag{2}$$

where $y \in I\!R^r$ denotes a vector of external coordinates. The number of degrees of freedom in Eq. (1) will be assumed to exceed the number of control inputs ($n > m$), which means that the system composed of Eqs. (1) and (2) constitutes an underactuated system.

A standard substitution of a new state variable, $x = (x_1, x_2) = (q, \dot{q})$, converts the system into an affine control system

$$\left.\begin{aligned} \dot{x} &= f(x) + G(x)u, \\ y &= h(x) \end{aligned}\right\} \tag{3}$$

with the drift vector and the control matrix defined as follows

$$f(x) = \begin{pmatrix} x_2 \\ -M^{-1}(x_1)F(x) \end{pmatrix}, \quad G(x) = \begin{bmatrix} 0_{n\times m} \\ M^{-1}(x_1)B(x_1) \end{bmatrix},$$

where $0_{n\times m}$ is an $n \times m$ zero matrix and matrices $M(x_1)$, $F(x)$, $B(x_1)$ come from Eq. (1).

3. Motion Planning Problem

Assume that the motion planning problem for the system (3) consists of the primary, proper motion planning task, and one or more secondary tasks of decreasing priorities. The objective of the motion planner is to solve all these tasks simultaneously, respecting their priorities. The motion planning task consists in determining a control function which drives the output of the system (3) to a desirable point, over a prescribed time interval. Additionally, the control function is supposed to solve all the subordinate tasks, however, if the solution of all subtasks is not possible, the tasks of higher priority will be preferred. The construction of the motion planning algorithm combines two ingredients: the generalized inverse of the Jacobian with task priority, based on ref. [8], and the endogenous configuration space approach.[5]

4. Algorithm Construction

In order to solve the motion planning problem, one needs to define an algorithm solving the primary, motion planning task, and then the way to incorporate the remaining tasks. As a result, a complete task-priority motion planning algorithm is obtained, able to solve the problem composed of several tasks. For making the derivation self-contained, in the next subsection some basic concepts referring to the endogenous configuration space approach will be introduced.

4.1. Endogenous configuration space

Following ref. [5], the endogenous configuration of the control system (3) is identified with an admissible control function $u(\cdot) \in L^2_m[0,T] = \mathcal{X}$ defined on a time interval $[0,T]$. The endogenous configuration space $\mathcal{X}$ is a Hilbert space equipped with the inner product

$$\langle u_1(\cdot), u_2(\cdot)\rangle_{\mathcal{X}} = \int_0^T u_1^T(t)R(t)u_2(t)\,\mathrm{d}t, \quad R(t) = R^T(t) > 0,$$

and the induced norm. To every endogenous configuration there corresponds a state trajectory $x(t) = \varphi_{x_0,t}(u(\cdot))$, where $\varphi_{x_0,t}(u(\cdot))$ denotes the flow of the system (3) initialized at x_0 and driven by $u(\cdot)$.

The accomplishment of a task as it depends on control is described by a task map

$$K_T(u(\cdot))\colon \mathcal{X} \to I\!R^s, \tag{4}$$

which transforms the endogenous configuration into a task space. By the differentiation of (4) the Jacobian is obtained

$$J_T(u(\cdot))v(\cdot) = \left.\frac{\mathrm{d}}{\mathrm{d}\alpha}\right|_{\alpha=0} K_T(u(\cdot) + \alpha v(\cdot)). \tag{5}$$

If the Jacobian is invertible, then there exists the Jacobian pseudo-inverse[5]

$$J_T^{\#}(u(\cdot)) : I\!R^s \to \mathcal{X}. \tag{6}$$

From the properties of the pseudo-inverse,[13] it follows that the map

$$P_T(u(\cdot)) = \mathrm{id}_{\mathcal{X}} - J_T^{\#}(u(\cdot))J_T(u(\cdot)), \tag{7}$$

where $\mathrm{id}_{\mathcal{X}}$ denotes the identity map, is a projection of the endogenous configuration space onto $\ker J_T(u(\cdot))$. Specific formulas for Eqs. (4)–(6) will be provided in the next subsections.

4.2. Jacobian motion planning algorithm

The construction of the task-priority algorithm begins with a computation of the Jacobian for the motion planning task, and its pseudo-inverse. The task map is in this case tantamount to the end point map of the control system (3),

$$\begin{aligned} &K_{x_0,T}(u(\cdot))\colon \mathcal{X} \to I\!R^r, \\ &K_{x_0,T}(u(\cdot)) = y(T) = h(\varphi_{x_0,T}(u(\cdot))), \end{aligned} \tag{8}$$

so it determines the output of the system (3) reached at time T under the control $u(\cdot)$. In accordance with (5), the Jacobian results from the differentiation of Eq. (8),

$$J_{x_0,T}(u(\cdot))v(\cdot) = \left.\frac{\mathrm{d}}{\mathrm{d}\alpha}\right|_{\alpha=0} K_{x_0,T}(u(\cdot)+\alpha v(\cdot)) = C(T)\int_0^T \Phi(T,s)B(s)v(s)\,\mathrm{d}s,$$

The computation of the Jacobian involves the linear approximation

$$\left.\begin{aligned} \dot\xi &= A(t)\xi + B(t)v, \\ \eta &= C(t)\xi, \end{aligned}\right\} \tag{9}$$

to the system (3) along a control-trajectory pair $(u(t), x(t))$, where the matrices $A(t)$, $B(t)$, $C(t)$ are defined as follows:

$$A(t) = \frac{\partial(f(x(t)) + G(x(t))u(t))}{\partial x},$$
$$B(t) = G(x(t)), \quad C(t) = \frac{\partial h(x(t))}{\partial x}.$$

The transition matrix $\Phi(t,s)$ of the linearized system satisfies the evolution equation $\frac{\partial\Phi(t,s)}{\partial t} = A(t)\Phi(t,s)$, with initial condition $\Phi(s,s) = I_n$.

Relying on a derivation accomplished in ref. [5], the Jacobian peudo-inverse

$$\left(J^{\#}_{x_0,T}(u(\cdot))\eta\right)(t) = B^T(t)\Phi^T(T,t)C^T(T)\mathcal{G}^{-1}_{x_0,T}(u(\cdot))\eta, \tag{10}$$

where $\mathcal{G}_{x_0,T}(u(\cdot))$ is the Gram matrix of the linear system (9), defined as

$$\mathcal{G}_{x_0,T}(u(\cdot)) = C(T)\int_0^T \Phi(T,s)B(s)B^T(s)\Phi^T(T,s)\,\mathrm{d}s\, C^T(T).$$

The Jacobian pseudo-inverse is well defined on condition that the Gram matrix in nonsingular.

Given the Jacobian pseudo-inverse (10), the following procedure defines the Jacobian pseudo-inverse motion planning algorithm. First, an initial control $u_0(\cdot) \in \mathcal{X}$ is chosen. If this control does not solve the motion planning problem, it is deformed to a smooth curve $u_\theta(\cdot) \in \mathcal{X}$, and the error

$$e(\theta) = K_{x_0,T}(u_\theta(\cdot)) - y_d,$$

is computed, where y_d denotes a desirable point in the task space of the system (3). A requirement that the error decreases exponentially at a decay rate $\gamma > 0$ leads to a Ważewski–Davidenko equation

$$J_{x_0,T}(u_\theta(\cdot))\frac{\mathrm{d}}{\mathrm{d}\theta}u_\theta(\cdot) = -\gamma e(\theta). \tag{11}$$

The Jacobian pseudo-inverse transforms this equation into a dynamic system

$$\frac{\mathrm{d}}{\mathrm{d}\theta}u_\theta(t) = -\gamma\left(J^{\#}_{x_0,T}(u_\theta(\cdot))\, e(\theta)\right)(t), \tag{12}$$

whose limit trajectory $u_d(t) = \lim_{\theta\to+\infty} u_\theta(t)$ provides a solution to the motion planning problem.

The basic system (12) can be extended to the form

$$\frac{\mathrm{d}}{\mathrm{d}\theta}u_\theta(t) = -\gamma\left(J^{\#}_{x_0,T}(u_\theta(\cdot))e(\theta)\right)(t) + P_{x_0,T}(u_\theta(\cdot))(\mu(\cdot))(t), \tag{13}$$

where $\mu(\cdot) \in \mathcal{X}$ and

$$P_{x_0,T}(u(\cdot)) = \mathrm{id}_{\mathcal{X}} - J^{\#}_{x_0,T}(u(\cdot))J_{x_0,T}(u(\cdot)).$$

It is easily checked that, since $P_{x_0,T}(u(\cdot))$ is a projection onto $\ker J_{x_0,T}(u(\cdot))$, the expression (13) satisfies the Eq. (11). The term $\mu(\cdot)$ denotes a direction of motion in the endogenous configuration space that will be further exploited in the construction of the task-priority algorithm.

4.3. Secondary task

As an example of the secondary task the control energy minimization is chosen. The corresponding task map

$$K_T(u(\cdot))\colon \mathcal{X} \to I\!R, \quad K_T(u(\cdot)) = \frac{1}{2}\int_0^T u^T(t)\sigma(t)u(t)\,\mathrm{d}t$$

that has the meaning of the weighted control energy, where $\sigma(t) = \sigma^T(t)$ denotes a weight matrix. According to Eq. (5), the Jacobian is computed as the differential

$$J_T(u(\cdot))v(\cdot) = \left.\frac{\mathrm{d}}{\mathrm{d}\alpha}\right|_{\alpha=0} K_T(u(\cdot)+\alpha v(\cdot)) = \int_0^T u^T(t)\sigma v(t)\,\mathrm{d}t.$$

The Jacobian pseudo-inverse for the secondary task arises as a solution $v(\cdot)$ of the Jacobian equation

$$J_T(u(\cdot))v(\cdot) = \int_0^T u^T(t)\sigma(t)v(t)\,\mathrm{d}t = \zeta, \qquad \zeta \in I\!R, \tag{14}$$

which minimize the integral

$$\frac{1}{2}\int_0^T v^T(t)v(t)\,\mathrm{d}t.$$

A standard solution procedure of this constrained optimization problem[14] involves the Lagrangian

$$\mathcal{L}(v(\cdot),\lambda) = \frac{1}{2}\int_0^T v^T(t)v(t)\,\mathrm{d}t + \lambda\left(\int_0^T u^T(t)\sigma(t)v(t)\mathrm{d}t - \zeta\right),$$

where $\lambda \in I\!R$ denotes a Lagrange multiplier. After some standard developments, the Jacobian pseudo-inverse for the secondary task is found as

$$\left(J^{\#}_T(u(\cdot))\zeta\right)(t) = \frac{\sigma(t)u(t)}{\|\sigma(\cdot)u(\cdot)\|^2}\zeta,$$

where $\|\sigma(\cdot)u(\cdot)\|^2 = \int_0^T u^T(t)\sigma^T(t)\sigma(t)u(t)\,\mathrm{d}t$. By repeating the same line of reasoning as for the primary task, the following Jacobian pseudo-inverse algorithm solving the secondary task is obtained

$$\frac{\mathrm{d}}{\mathrm{d}\theta}u_\theta(t) = -\gamma\left(J_T^{\#}(u_\theta(\cdot))e(\theta)\right)(t) + P_T(u(\cdot))(\mu(\cdot))(t),$$

where $P_T(u_\theta(\cdot)) = \mathrm{id}_{\mathcal{X}} - J_T^{\#}(u(\cdot))J_T(u(\cdot))$ stands for the projection onto $\ker J_T(u(\cdot))$, $\mu(\cdot)$ has the same meaning as in Eq. (13), and the error

$$e(\theta) = K_T(u_\theta(\cdot)) = \frac{1}{2}\int_0^T u_\theta^T(t)\sigma(t)u_\theta(t)\,\mathrm{d}t.$$

By design, the algorithm converges exponentially with the rate $\gamma > 0$. A construction of the algorithm solving a secondary task of the control energy minimization may serve as an example. Other secondary tasks can be taken into account analogously.

4.4. *Task-priority Jacobian inverse algorithm*

In this subsection two previously devised algorithms will be combined into a task-priority motion planning algorithm including two subtasks, and next this algorithm will be generalized to more than two subtasks. Let $(S_1, \ldots, S_l)$ denotes a collection of subtasks, ordered with decreasing priority. With every task S_i a corresponding task map $K_{iT}\colon \mathcal{X} \to I\!R^{s_i}$ (Eq. (4)) is associated, as well as, a task Jacobian $J_{iT}(u(\cdot))\colon \mathcal{X} \to I\!R^{s_i}$ (Eq. (5)), a Jacobian pseudo-inverse $J_{iT}^{\#}(u(\cdot))\colon I\!R^{s_i} \to \mathcal{X}$ (Eq. (6)), and the error $e_i(u(\cdot)) = K_{iT}(u(\cdot)) - y_{id}$, where y_{id} stands for the desirable value of the map $K_{iT}(u(\cdot))$. This being so, the ith task may be described by a quadruple $S_i = (K_{iT}, J_{iT}J_{iT}^{\#}, e_i)$. As it was mentioned in the previous subsection, the variable $\mu(\cdot)$ in Eq. (13) can be replaced by a term related to the accomplishment of a lower priority task. The Jacobian equation for the ith subtask takes the form

$$\frac{\mathrm{d}e_i(\theta)}{\mathrm{d}\theta} = J_{iT}(u_\theta(\cdot))\frac{\mathrm{d}u_\theta(\cdot)}{\mathrm{d}\theta}, \qquad i = 1, 2, \ldots, l.$$

For the task S_i the pseudo-inverse $J_{iT}^{\#}(u(\cdot))$ defines a general Jacobian pseudo-inverse algorithm

$$\frac{\mathrm{d}u_\theta(\cdot)}{\mathrm{d}\theta} = -\gamma_i J_{iT}^{\#}(u_\theta(\cdot))e_i(\theta) + P_{iT}(u_\theta(\cdot))\mu_i(\cdot), \quad i = 1, 2, \ldots, l, \tag{15}$$

where $P_{iT}(u(\cdot)) = \mathrm{id}_{\mathcal{X}} - J_{iT}^{\#}(u(\cdot))J_{iT}(u(\cdot))$ is the projection onto $\ker J_{iT}(u(\cdot))$, and $\mu_i(\cdot) \in \mathcal{X}$. From an axiom of the pseudo-inverse[13],

$$J_{iT}^{\#}(u(\cdot))J_{iT}(u(\cdot))J_{iT}^{\#}(u(\cdot)) = J_{iT}^{\#}(u(\cdot)) \tag{16}$$

it follows directly that every operator $P_{iT}(u(\cdot))$ is idempotent, i.e.,

$$P_{iT}^2(u(\cdot)) = P_{iT}(u(\cdot))P_{iT}(u(\cdot)) = P_{iT}(u(\cdot)).$$

Now, following the route taken in ref. [8], the algorithm comprising two subtasks can be derived. Note that, differently to the original setting in ref. [8], the dynamic system (15) evolves in the infinite-dimensional endogenous configuration space. For each pair of tasks (say for tasks S_1 and S_2), the identity (15) yields

$$\begin{aligned}&-\gamma_2 J_{2T}^{\#}(u_\theta(\cdot))e_2(\theta) + P_{2T}(u_\theta(\cdot))\mu_2(\cdot)\\ &\quad = -\gamma_1 J_{1T}^{\#}(u_\theta(\cdot))e_1(\theta) + P_{1T}(u_\theta(\cdot))\mu_1(\cdot).\end{aligned} \tag{17}$$

Next, each side of Eq. (17) is multiplied from the left by the operator $P_{1T}(u(\cdot))$. The property (16) implies that $P_{iT}(u(\cdot))J_{iT}^{\#}(u(\cdot)) = 0$, so, by the idempotency of the projection, the identity (17) transforms to the form

$$\begin{aligned}&-\gamma_2 P_{1T}(u_\theta(\cdot))J_{2T}^{\#}(u_\theta(\cdot))e_2(\theta)\\ &\quad + P_{1T}(u_\theta(\cdot))P_{2T}(u_\theta(\cdot))\mu_2(\cdot) = P_{1T}(u_\theta(\cdot))\mu_1(\cdot).\end{aligned} \tag{18}$$

After the substitution of Eq. (18) into (15) for $i = 1$, the following motion planning algorithm for two subtasks is obtained

$$\begin{aligned}\frac{\mathrm{d}u_\theta(\cdot)}{\mathrm{d}\theta} &= -\gamma_1 J_{1T}^{\#}(u_\theta(\cdot))e_1(\theta) - \gamma_2 P_{1T}(u_\theta(\cdot))J_{2T}^{\#}(u_\theta(\cdot))e_2(\theta)\\ &\quad + P_{1T}(u_\theta(\cdot))P_{2T}(u_\theta(\cdot))\mu_2(\cdot).\end{aligned}$$

If there are only two subtasks, then $\mu_2(\cdot) = 0$, otherwise the term $\mu_2(\cdot)$ will enable the inclusion into the algorithm of the next subtask.

By repeating the presented line of reasoning, a recurrent formula of the task-priority motion planning algorithm including l subtasks is obtained

$$\frac{\mathrm{d}u_\theta(\cdot)}{\mathrm{d}\theta} = -\sum_{i=1}^{l}\gamma_i\left(\prod_{j=0}^{i-1}P_{jT}(u_\theta(\cdot))\right)J_{iT}^{\#}(u_\theta(\cdot))e_i(\theta),$$

where $P_{0T}(u_\theta(\cdot)) = \mathrm{id}_{\mathcal{X}}$.

4.5. *Computational aspects*

For a practical computer implementation of the motion planning algorithm, it is necessary to employ a finite dimensional parametrization of the endogenous configuration space, and to use a discrete version of the algorithm instead of the continuous one. In order to get the finite dimensional representation, the control function of the system (3) should be chosen in the form of a truncated orthogonal series of given order k, composed of certain basic functions $\psi_i(t)$ belonging to an orthogonal basis of $\mathcal{X}$

$$u_i(\lambda, t) = \sum_{j=0}^{k}\lambda_{ij}\psi_j(t) = \Psi(t)\lambda, \qquad i = 1, 2, \ldots, m, \tag{19}$$

where $\psi_j(t)$ denotes the jth function from the basis. λ collects the gain coefficients and is defined as $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_k)$, $\lambda_j = (\lambda_{0j}, \lambda_{2j}, \ldots, \lambda_{mj})$, and matrix $\Psi(t) = [I_m\psi_0(t), I_m\psi_1(t), \ldots, I_m\psi_k(t)]$ is the block matrix built of the basic functions. The choice of the truncation order

k should satisfy the following inequality

$$(k+1)m \geq \sum_{i=1}^{l} s_i. \tag{20}$$

The parametrization mentioned above reduces the endogenous configuration space to a finite dimensional space $\widetilde{\mathcal{X}} = I\!R^{(k+1)m}$. Consequently, the task map (4) becomes finite dimensional

$$\widetilde{K}_T(\lambda) = K_T(u(\lambda,\cdot))\colon \widetilde{\mathcal{X}} \to I\!R^s,$$

furthermore, also the Jacobian and the Jacobian pseudo-inverse become finite dimensional operators represented by suitable matrices $\widetilde{J}_T(\lambda) = J_T(u(\lambda,\cdot))$ and $\widetilde{J}_T^{\#}(\lambda) = \widetilde{J}_T^T(\lambda)(\widetilde{J}_T(\lambda)\widetilde{J}_T^T(\lambda))^{-1}$.

The whole derivation from the previous section remains valid in the finite dimensional setting, leading finally to a finite dimensional and discrete version of the task-priority motion planning algorithm

$$\lambda_{\theta+1} = \lambda_\theta - \sum_{i=1}^{l} \gamma_i \left(\prod_{j=0}^{i-1} \widetilde{P}_{jT}(\lambda_\theta) \right) \widetilde{J}_{iT}^{\#}(\lambda_\theta)\widetilde{e}_i(\theta),$$

where $\theta = 0, 1, \ldots$, the matrix $\widetilde{P}_{jT}(\lambda) = I_{(k+1)m} - \widetilde{J}_{jT}^{\#}(\lambda)\widetilde{J}_{jT}(\lambda)$ is a projection onto $\ker \widetilde{J}_{jT}(\lambda)$, $I_{(k+1)m}$ is a $(k+1)m$ dimensional identity matrix, $\widetilde{e}_i(\theta) = \widetilde{K}_{iT}(\lambda_\theta) - y_{id}$ denotes the ith task error.

5. Application to Container Ship

An assessment of performance and efficiency of the task-priority motion planning algorithm will be based on computer simulations. As a test bed a model of the container ship has been chosen, provided in ref. [4]. This section presents the model, tunes the algorithm to the ship model, and finally reports on results of computer simulations in the *Matlab* environment. The simulations will reveal that the task-priority motion planning algorithm outperforms the simple single-task algorithm.

5.1. Ship dynamics

In this subsection the model of a container ship[4] is considered. The ship is equipped with a rudder enabling a change of the ship's orientation, and with a forward thrust propeller controlling the ship's velocity. Suppose that the vector $\eta = (n, e, \phi) \in I\!R^3$ denotes the North–East position and the yaw angle of the ship, and that the vector $\nu = (u, v, r) \in I\!R^3$ is the velocity vector expressed in the body frame. Then, the kinematics equation can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t}\eta = R(\phi)\nu, \qquad R(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

where $R(\phi)$ is the rotation matrix around the Z-axis. The ship's dynamics equation, normalized using the Prime system

Table I. Numerical values of coefficients of the ship's dynamics.

$m' = 0.00792$	$m'_x = 0.000238$	$m'_y = 0.007049$
$I'_z = 0.000456$	$J'_z = 0.000419$	$X'_{uu} = -0.0004226$
$Y'_v = -0.0116$	$Y'_r = 0.00242$	$Y'_\delta = -0.002578$
$N'_v = -0.0038545$	$N'_r = -0.00222$	$N'_\delta = 0.00126$
$t_d = 0.175$	$\rho = 1000\,[\mathrm{kg/m^3}]$	$L = 175\,[\mathrm{m}]$

of SNAME,[4] with respect to the body-fixed frame is

$$M(\nu)\dot{\nu} + N(\nu)\nu = B(\nu)u, \tag{21}$$

where $M(\nu)$ denotes a symmetric and positive definite inertia matrix, and, depending on the assumed model accuracy, the term $N(\nu)\nu$ expresses centrifugal, Coriolis, hydrodynamics, or friction torques/forces. $B(\nu)$ is a control matrix and $u = (\Gamma, \delta) \in I\!R^2$ denotes the control input, where Γ stands for the propeller forward thrust and δ is the turn angle of the rudder. The matrices appearing in Eq. (21) are defined in the following way:[4]

$$M(\nu) = \begin{bmatrix} \dfrac{L(m'+m'_x)}{U^2} & 0 & 0 \\ 0 & \dfrac{L(m'+m'_y)}{U^2} & 0 \\ 0 & 0 & \dfrac{L^2(I'_z+J'_z)}{U^2} \end{bmatrix},$$

$$U = \sqrt{u^2+v^2},$$

$$N(\nu) = \begin{bmatrix} \dfrac{-X'_{uu}|u/U|}{U} & 0 & \dfrac{-L(m'+m'_y)v/U}{U} \\ 0 & \dfrac{-Y'_v}{U} & \dfrac{-L(Y'_r+(m'+m'_x)u/U)}{U} \\ 0 & \dfrac{-N'_v}{U} & \dfrac{-LN'_r}{U} \end{bmatrix},$$

$$B = \begin{bmatrix} \dfrac{2(1-t_d)}{\rho U^2 L^2} & 0 \\ 0 & Y'_\delta \\ 0 & N'_\delta \end{bmatrix}.$$

Numerical values of the above coefficients have been collected in Table I. The coefficients m', m'_x, and m'_y are dimensionless and refer to the mass of the ship and to the added mass effect. Symbols I'_z, J'_z denote normalized inertia torques of the ship, and torques of the added mass. The quantities X'_{uu}, Y'_v, Y'_r, Y'_δ, N'_v, N'_r, and N'_δ are dimensionless hydrodynamic coefficients. t_d is the propeller forward thrust ratio, ρ denotes the water density, and L is the length of the ship. In the coordinates $x = (x_1, x_2) = (\eta, \nu) \in I\!R^6$ the ship's dynamics Eq. (21) can be written as

$$\left.\begin{aligned} \dot{x}_1 &= R(x_{13})x_2, \\ \dot{x}_2 &= M^{-1}(\nu)(B(x_2)u - N(x_2)x_2), \end{aligned}\right\} \tag{22}$$

where x_{13} denotes the third coordinate of the vector x_1 (the orientation angle ϕ). The output function is set to

$$y = h(x) = x_1 = \eta, \tag{23}$$

where $y \in I\!R^3$ denotes a vector of task coordinates. Equations (22) and (23) define an affine control system with output

$$\left.\begin{aligned} \dot{x} &= f(x) + G(x)u, \\ y &= h(x), \end{aligned}\right\} \tag{24}$$

where

$$f(x) = \begin{pmatrix} R(x_{13})x_2 \\ M^{-1}(x_2)N(x_2)x_2 \end{pmatrix}, \quad G(x) = \begin{bmatrix} 0_{3\times 2} \\ M^{-1}(x_2)B(x_2) \end{bmatrix}.$$

The control system (24) contains the order 2 nonholonomic constraints, resulting from the dynamics of the system. Because the dimension of the task vector $r = 3$ is greater than the number of independent control inputs $m = 2$, this ship model belongs to the class of underactuated systems. In the next subsections a motion planning algorithm for the model (24) will be constructed.

5.2. *Motion planning problem*

The motion planning problem has been outlined in Section 3, so only the primary and the secondary task should be defined. The motion planning problem amounts to finding the control function $u(t)$ that, given the initial configuration $x(0)$ and the control horizon $[0, T]$, drives the ship to a desirable point $y(T) = y_d$ in the task space. Additionally, it is required that the control energy should be minimal. This formulation involves two subtasks

$$x(0) \xrightarrow{u_d(\cdot)} y_d, \tag{25}$$

$$\int_0^T u_d^T(t)\sigma(t)u_d(t)\,\mathrm{d}t = \min_{u(\cdot)} \int_0^T u^T(t)\sigma(t)u(t)\,\mathrm{d}t, \tag{26}$$

where the task (25) has been given a higher priority than (26).

5.3. *Ship motion planning algorithm*

In order that provide a solution to the motion planning problem it is necessary to construct the maps (4), the Jacobians (5), the Jacobian pseudo-inverses (6), and define the error formulas for both subtasks. The infinite dimensional version of these objects was introduced in Subsections 4.2 and 4.3. For the purpose of computer simulations, a finite dimensional version of the task-priority algorithm is needed. A control functions parametrization of the system (24) will be accomplished in accordance with (19), using truncated Fourier series. For the ship's model the matrix $\Psi(t)$ in (19) is set as

$$\Psi(t) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & \sin\omega t & \cos\omega t & \sin 2\omega t \end{bmatrix},$$

so $\dim\lambda = 5$, and the condition (20) holds. The proper motion planning task is specified as $S_1 = (\widetilde{K}_{1\,x_0,T},\ \widetilde{J}_{1\,x_0,T},\ \widetilde{J}^{\#}_{1\,x_0,T},\ \widetilde{e}_1)$, analogously, the control energy minimization task $S_2 = (\widetilde{K}_{2\,T},\ \widetilde{J}_{2\,T},\ \widetilde{J}^{\#}_{2\,T},\ \widetilde{e}_2)$. To construct the algorithm, all constitutive elements of these tasks should be defined.

For the task (S_1) the finite dimensional Jacobian is defined as

$$\widetilde{J}_{1\,x_0,T}(\lambda) = C(T)\int_0^T \Phi(T,s)B(s)\Psi(s)\,\mathrm{d}s,$$

so the Jacobian pseudo-inverse becomes

$$\widetilde{J}^{\#}_{1\,x_0,T}(\lambda) = \widetilde{J}^{T}_{1\,x_0,T}(\lambda)\left(\widetilde{J}_{1\,x_0,T}(\lambda)\widetilde{J}^{T}_{1\,x_0,T}(\lambda)\right)^{-1}.$$

Concerning the task S_2, the finite dimensional task map is

$$\widetilde{K}_{2\,T}(\lambda) = K_{2\,T}(\Psi(\cdot)\lambda) = \frac{1}{2}\int_0^T (\Psi(t)\lambda)^T\sigma(t)(\Psi(t)\lambda)\,\mathrm{d}t$$
$$= \frac{1}{2}\lambda^T \underbrace{\int_0^T \Psi^T(t)\sigma(t)\Psi(t)\,\mathrm{d}t}_{Q(T)}\,\lambda = \frac{1}{2}\lambda^T Q(T)\lambda,$$

yielding the finite dimensional Jacobian

$$\widetilde{J}_{2\,T}(\lambda) = \frac{\mathrm{d}}{\mathrm{d}\lambda}\widetilde{K}_{2\,T}(\lambda) = Q(T)\lambda,$$

and the Jacobian pseudo-inverse

$$\widetilde{J}^{\#}_{2\,T}(\lambda) = \widetilde{J}^{T}_{2\,T}(\lambda)\left(\widetilde{J}_{2\,T}(\lambda)\widetilde{J}^{T}_{2\,T}(\lambda)\right)^{-1}.$$

Finally, a finite dimensional and discrete version of the task-priority motion planning algorithm is derived

$$\lambda_{\theta+1} = \lambda_\theta - \gamma_1\widetilde{J}^{\#}_{1\,x_0,T}\widetilde{e}_1(\lambda_\theta) - \gamma_2(I_5 - \widetilde{J}^{\#}_{1\,x_0,t}(\lambda_\theta)\widetilde{J}_{1\,x_0,T}(\lambda_\theta)) \times \widetilde{J}^{\#}_{2\,x_0,T}(\lambda_\theta)\widetilde{e}_2(\lambda_\theta), \tag{27}$$

where the errors are defined as

$$\widetilde{e}_1(\lambda_\theta) = K_{1\,x_0,T}(u(\lambda_\theta,\cdot)) - y_d \ \text{ and } \ \widetilde{e}_2(\lambda_\theta) = K_{2\,T}(u(\lambda_\theta,\cdot)).$$

5.4. *Simulations results*

As a result, the algorithm (27) returns the control coefficients λ determining the required control function. For the purpose of the numeric implementation it has been chosen $\gamma_1 = 0.08$ and $\gamma_2 = 0.04$. The choice of a constant weight matrix $\sigma = \mathrm{diag}\{0, 1\}$ means the minimization of energy of the second control input (the rudder angle δ). In the real ships, the rudder angle is limited to $\delta \in (-10°, 10°) \approx (-0.175, 0.175)$. The motion planning problem introduced in the Subsection 5.3 is regarded as solved when the ship reaches the desirable point in the task space with the error below $\|e_1\| \le 10^{-6}$ (subtask S_1), and the value of the control δ remains inside the assumed bounds (subtask S_2). Because of the task S_2 has been defined as the energy minimization of the control function δ, after finishing the computations, it should be checked, if $\delta(t)$ lies inside the bounds.

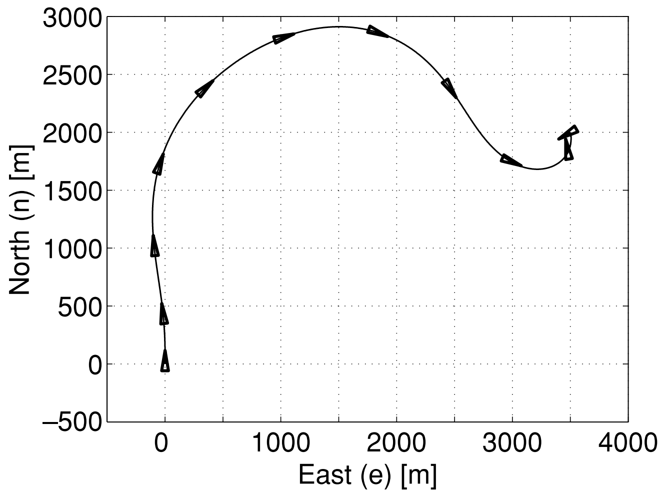

Fig. 1. Single-task algorithm (only S_1).

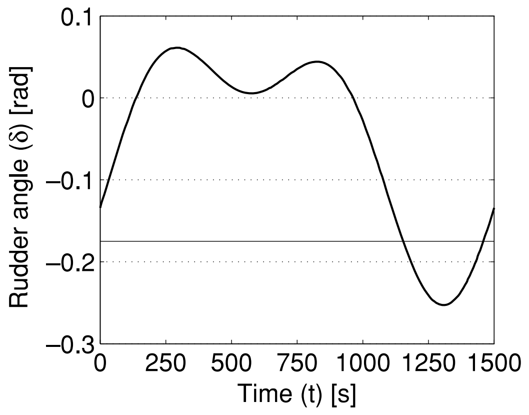

Fig. 2. Task-priority algorithm (both S_1 and S_2).

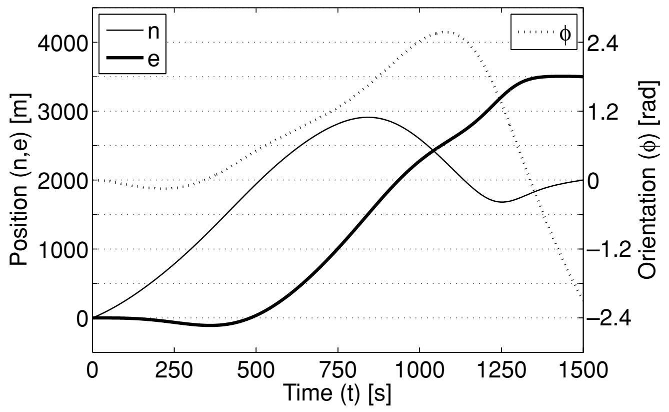

Fig. 3. Single-task algorithm (only S_1).

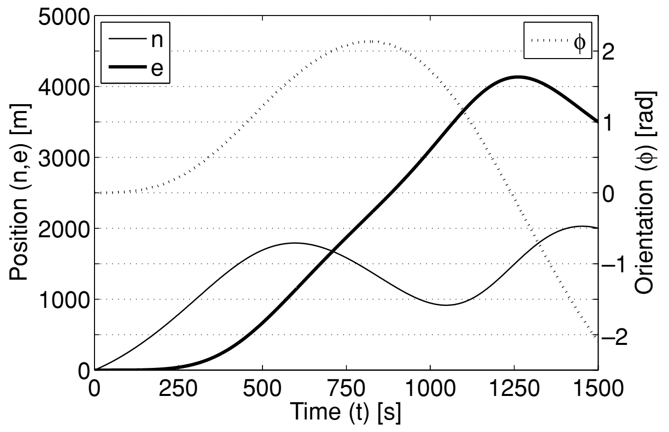

Fig. 4. Task-priority algorithm (both S_1 and S_2).

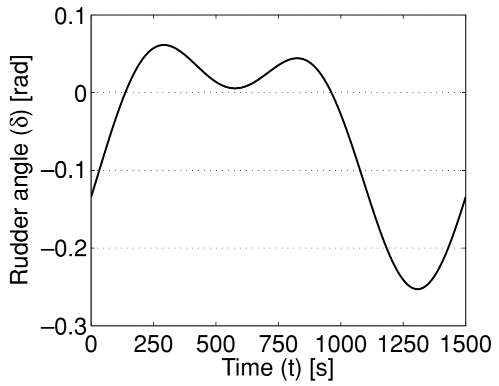

Fig. 5. Single-task algorithm (only S_1).

The simulation results are presented below. For comparison the results of all two algorithms: the single-task algorithm (without the task S_2), and task-priority algorithm (27) are revealed. The simulations started from the initial coefficient vector $\lambda_0 = (300000, 0, 0.1, 0, 0)$, the initial position of the ship is $x_1(0) = (0, 0, 0)$, and the initial velocity $x_2(0) = (2, 0, 0)$. The desirable task space point $y(T) = x_1(T) = (2000, 3500, -2\pi/3)$ should be reached in time $T = 1500$ s. For both algorithms the solutions have been produced in less than 500 steps.

In the Figs. 1 and 2 the ship's routes are visualized. Taking into account the motion strategy, both shapes of computed ship trajectories are similar. However, it can be observed that the trajectory returned from the single-task algorithm (fig. 1) cannot be applied to the real ship due to physical motion constraints, because the ship makes unrealistic turn near the end point. The profile of the ship in these figures has been marked every 150 s.

In the Figs. 3 and 4 the position and orientation trajectories of the ship are depicted. As it can be checked, both the algorithms solve the primary task (S_1) correctly, and at the end of the time horizon T the ship reaches the desirable point in the task space.

Note that, due to a specific selection of the matrix $\Psi(t)$ in Eq. (19), the first component of the control function, the thrust forward propeller Γ, remains constant at the value $\Gamma = 300$ kN. The rudder angle control function δ is depicted in the Figs. 5 and 6. Additionally, in the Fig. 5 the control bounds have been shown. As can be seen, the control function computed by the single-task algorithm violates the bounds. On the contrary, the task-priority algorithm keeps the control function inside the assumed bounds. It is necessary to mention, that the task-priority algorithm deals with subtasks in order of its priorities. So, it is possible that the second subtask will not be accomplished (the control exceeds the bounds).

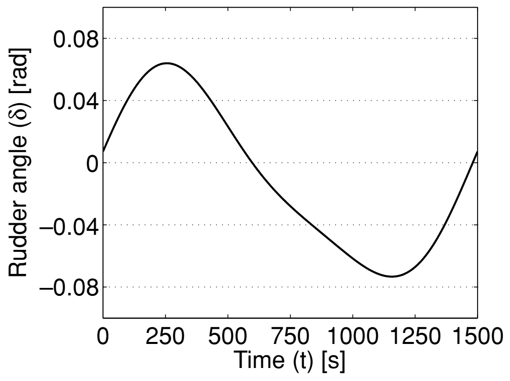

Fig. 6. Task-priority algorithm (both S_1 and S_2).

6. Conclusion

This paper provides a task-priority motion planning algorithm for underactuated robotic systems. The algorithm makes a fusion of the task-priority control of the holonomic manipulators, and of the Jacobian motion planning algorithms elaborated within the endogenous configuration space approach. The new algorithm solves the motion planning problem consisting of the primary motion planning task and of a collection of secondary tasks of decreasing priorities. Because the new algorithm computes a control function which solves the tasks with respect to the order of their priorities, it may happen that a subtask with lower priority will not be satisfied, exactly as in the case of the task-priority algorithm for holonomic manipulators.[7] A derivation of the algorithm for two subtasks has been made in detail, then a recurrent formula for more than two subtasks has been given. For the purpose of the derivation the secondary task has been chosen as the control energy minimization, however, an inclusion of other subtasks, like the obstacle avoidance, singularity avoidance, etc. presents no essential difficulties.

Performance of the algorithm has been illustrated with the container ship model. Particular attention has been paid to a comparison between the single–task algorithm (a proper motion planning using the Jacobian pseudo-inverse) and the task-priority algorithm. As might be expected, the sole motion planning task can be solved both by the single-task algorithm, as well as, by the task-priority algorithm. However, since the task-priority algorithm includes the subtask of control energy minimization, the amplitude of the rudder angle is smaller than in the solution delivered by the single-task algorithm. Since in real ships the maximum angle of the rudder is limited, the new algorithm certainly provides a more practical solution. The comparison of the Figs. 5 and 6 also shows that the control effort of the task-priority algorithm is smaller than of the single-task algorithm. In addition, the ship's route resulting from the task-priority algorithm outperforms the no task-priority solution.

The authors are indebted to anonymous reviewers whose comments resulted in improvement of the content and the readability of this paper.

Acknowledgments

This research has been supported in part by the Foundation for Polish Science and in part by the Wroclaw University of Technology under a statutory grant.

References

1. M. W. Spong, "Underactuated Mechanical Systems," **In**: *Control Problems in Robotics and Automation (Lecture Notes in Control and Information Sciences 230)* (B. Siciliano and K. Valavanis, eds.) (London, Springer-Verlag, 1997) pp. 135–150.
2. A. De Luca, S. Iannitti, R. Mattone and G. Oriolo, "Control Problems in Underactuated Manipulators," *Proceedings IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2001)*, Como, Italy (July 2001) pp. 855–861.
3. A. Shiriaev, A. Robertsson, P. Pacull and T. I. Fossen, "Motion Planning and Its Feedback Stabilization for Underactuated Ships: Virtual Constraints Approach," *Proceedings 16th IFAC World Congress*, Prague, Czech Republic (July 2005) pp. 67–72.
4. T. I. Fossen, *Guidance and Control of Ocean Vehicles* (J. Wiley, New York, 1994).
5. K. Tchoń and J. Jakubiak, "Endogenous configuration space approach to mobile manipulators: A derivation and performance assessment of Jacobian inverse kinematics algorithms," *Int. J. Control* **76**, 1387–1419 (Sep. 2003).
6. K. Tchoń, J. Jakubiak and K. Zadarnowska, "Mobile Manipulators with Affine Control System Representation," **In**: *Cybernetics of Robotic Systems* (K. Tchoń, ed.) (WKiŁ, Warsaw, PL, 2004) pp. 75–116.
7. Y. Nakamura, *Advanced Theoretical Robotics: Redundancy and Optimization* (Prentice Hall, New Jersey, NJ, 1991).
8. S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Trans. Robot. Autom.* **13**, 398–410 (Jun. 1997).
9. A. Ratajczak and K. Tchoń, "Control of Underactuated Robotic Manipulators: An Endogenous Configuration Space Approach," *Proceedings IEEE International Conference on Methods and Models in Automation and Robotics (MMAR 2007)*, Szczecin, Poland (2007) pp. 985–990.
10. A. Ratajczak, J. Karpińska and K. Tchoń, "Task Priority Motion Planning Algorithm for an Ocean Ship: The Endogenous Configuration Space Approach," *Tenth National Conference on Robotics*, Piechowice, Poland (2008) (in Polish).
11. J. Karpińska, Models and Control Algorithms of Underwater Robots *Master's Thesis* (Wrocław, Poland: Wrocław University of Technology, 2007) (in Polish).
12. G. Antonelli, *Underwater Robots – Motion and Force Control of Vehicle-Manipulator Systems* (Springer-Verlag, Berlin, 2003).
13. A. Ben-Israel and T. N. Greville, *Generalized Inverses Theory and Applications* (Springer-Verlag, New York, 2003).
14. D. G. Luenberger, *Optimization by Vector Space Methods* (J. Wiley, New York, 1999).