

Mechanical Design and Control of the new 7-DOF CyberMotion Simulator

Carlo Masone, Paolo Robuffo Giordano, Heinrich H. Bühlhoff

Abstract—This paper describes the mechanical and control design of the new 7-DOF CyberMotion Simulator, a redundant industrial manipulator arm consisting of a standard 6-DOF anthropomorphic manipulator plus an actuated cabin attached to the end-effector. Contrarily to Stewart platforms, an industrial manipulator offers several advantages when used as motion simulator: larger motion envelope, higher dexterity, and possibility to realize any end-effector posture within the workspace. In addition to this, the new actuated cabin acts as an additional joint and provides the needed kinematic redundancy to cope with the robot actuator and joint range constraints, which in general can significantly deteriorate the desired motion cues the robot is reproducing. In particular, we will show that, by suitably exploiting the redundancy better results can be obtained in reproducing sustained acceleration cues, a relevant problem when implementing vehicle simulators.

I. INTRODUCTION

The realization of realistic immersions in virtual environments is an active research field in robotics [1]. Usually, in robotics-related fields, immersion may refer to providing visual/haptic sensory cues to a human operator to enhance his situational awareness of remote locations, to allow him interacting with remote environments, or to train him in specific tasks [2], [3]. In this respect, when dealing with training procedures for pilots, an additional relevant cue is represented by the sense of self-motion that usually completes the haptic/visual information to improve the immersiveness. In this context, simulators of vehicle motion represent one major application of interaction with a virtual environment for training purposes [4], but also for educating and entertaining as reported in [5].

Usually, motion simulators are based on fully actuated hexapods (Stewart platforms). Although with motion capabilities in 6 degrees of freedom (DOF), these platforms suffer from their limited workspace and the impossibility to achieve large linear and angular displacements and rates because of the the closed chain nature of their actuation system. As a possible improvement to this design, a novel motion platform based on a serial manipulator arm carrying a cabin on the end-effector is also gaining ground in the community because of its higher dexterity, larger motion envelope, the possibility to realize any end-effector posture within the workspace, and the ability to displace heavy loads with large accelerations and velocities [6], [7], [8],

[9], [10]. This is particularly the case of the CyberMotion Simulator, a 6-DOF anthropomorphic robot arm based on the commercial KUKA Robocoaster [11] for which a complete control framework for simulating general vehicle dynamics was recently presented and experimentally validated in [12], [13]. Compared to more advanced designs that have recently appeared in the motion simulation community [14], [15], the CyberMotion Simulator presents a good tradeoff between development/maintenance costs and motion capability.

Exploiting a serial manipulator as motion platform poses, of course, several challenges: first, suitable inverse kinematics schemes must be conceived to deal with an unpredictable and arbitrary desired cabin motion, generated online as a function of the (unpredictable) user inputs to the simulated vehicle. At the same time, one has to cope with the typical manipulator constraints such as joint limits and actuator saturations. Second, the design of washout filters, and in general of the whole *motion cueing* block [16], [17], [18], [19], must be tailored to the specific motion envelope of a serial manipulator. In [12], [13], a general solution to these problems is discussed but only validated for the particular case of the CyberMotion Simulator, i.e., a 6-DOF anthropomorphic manipulator with no redundancy w.r.t. a general Cartesian task. It is clear that a major limitation of such a system is represented by the limited range of the joints: whenever a joint reaches its limit during the operation, the reproduction of a desired motion can be heavily distorted because of the partial loss of mobility of the robot, thus inducing false cues on the pilot onboard the cabin.

In this respect, the goal of this paper is to present a novel extension of the CyberMotion Simulator that features an actuated cabin acting as an additional seventh *joint* for the whole system, thus granting a consistent increase of mobility thanks to the introduced redundancy. This redundancy can be exploited online to optimize the robot configuration and avoid singularities/joint limits, with the latter being particularly critical for the reproduction of the desired motion cues as explained before.

The paper is organized as follows: Sect. II describes the whole system (manipulator arm + cabin) and summarizes the main mechanical properties. Sect. III then focuses on the cabin mechanical design and kinetic analysis. Indeed, the cabin presents a non-trivial mechanical design because of its actuation system: it can switch from being a linear/rotation/hybrid joint during the motion. In Sect. III, we will show how to handle all the possible cases both for the forward and inverse kinematics analysis. After this, Sect. IV focuses on the control design for the whole system:

C. Masone, P. Robuffo Giordano, and H. H. Bühlhoff are with the Max Planck Institute for Biological Cybernetics, Spemannstraße 38, 72076 Tübingen, Germany.

H. H. Bühlhoff is also with the Department of Brain and Cognitive Engineering, Korea University, Anamdong, Seongbuk-gu, Seoul, 136-713 Korea.

here, an extension of the framework introduced in [12], [13] to deal with the redundant case is discussed. Finally, Sect. V reports simulation results showing the improvements of the novel manipulator/cabin design in reproducing sustained acceleration cues, as the ones typical of race cars. Reproducing high sustained accelerations was not easily achievable with the original CyberMotion Simulator setup because of the aforementioned joint limit constraints. We will show that, by exploiting the redundancy offered by the new cabin joint, better results can be obtained for this kinds of situations. Section VI concludes the paper and discusses future directions.

II. PRELIMINARIES

The CyberMotion Simulator is based on the standard six-joint anthropomorphic robot arm KUKA Robocoaster (a modified KR-500 industrial robot with a 500 [Kg] payload), which was originally designed for use in amusement parks. This setup, together with a seat rigidly attached to the robot end-effector, has been used to simulate the motion of a race car [12], [13], and of a helicopter at the ILA Berlin air show 2010 [20].

In order to improve the quality of the simulated experience, we designed a novel actuated cabin (Fig. 1) that will replace the original fixed seat. The cabin consists of two main parts: a $1.6 \times 1.8 \times 1.9$ [m] closed shell gondola that encapsulates the seat, and a rigid flange (the dashed red box in Fig. 1) attached to the end-effector of the robot. The two parts are connected to each other by two identical and parallel rails, fixed to the back of the gondola, and bound to move along a metal guide on the flange. The motion is provided by two servomotors capable of 328 [Nm] output torque. With respect to the previous setup, the new cabin extends the motion capability of the robot and also eliminates the unwanted external cues from the environment thanks to the closed shell design. Moreover, the gondola is equipped with two projectors for stereo visualization and can mount different input devices, in particular, a full force-feedback helicopter control device composed by a cyclic stick, a collective stick and actuated pedals, and a force-feedback steering wheel and pedal module. These input devices have force sensors that improve the accuracy of the system. The communication between the onboard control electronics and the control computer is via a CAN Bus running at 1 [kHz].

The robot joint actuation is implemented by a low-level controller that realizes a given joint velocity command at a fast rate, so that one can disregard any dynamical issue and consider joint velocities as actual control inputs, as classically done within the *kinematic control* framework [21]. Let $q = [q_M^T \ q_C]^T \in \mathbb{R}^7$ be the joint configuration vector, where $q_M = [q_1 \ \dots \ q_6] \in \mathbb{R}^6$ is the joint vector of the original KUKA Robocoaster and $q_C \in \mathbb{R}$ will be used afterwards to describe the configuration of the cabin, which will be treated as an additional seventh joint. The low-level controller accepts joint increment commands $\Delta q_k = q(t_k + 1) - q(t_k)$ as inputs, and returns the measured joint configuration $q(t_k)$ as output at a rate $T_s = 0.012$ [s].

	q_1	q_2	q_3	q_4	q_5	q_6	$q_7 = q_C$
q_{min}	-130	-128	-45	-180	-58	-180	0
q_{max}	130	-48	92	180	58	180	1.7317
\dot{q}_{max}	69	57	69	76	76	120	0.34
\ddot{q}_{max}	98	70	128	33	95	77	0.6

TABLE I: Joint range, velocity and acceleration limits of the 7-DOF manipulator with actuated cabin. Entries q_1 to q_6 are expressed in degrees, q_7 in meters.

Therefore, we model the robot as a single integrator

$$\dot{q} = u, \quad (1)$$

where $u \in \mathbb{R}^7$ is the commanded joint velocity. All the higher-lever control schemes will be built on top of (1).

The joint limitations of the robot (limited range, maximum velocity and maximum acceleration) are reported in Table I. The kinematic model of the actuated cabin will be described in the following section.

III. CABIN KINEMATICS

Let $\mathcal{F}_F : \{O_F; \vec{X}_F; \vec{Y}_F; \vec{Z}_F\}$ be a frame fixed to the center of the flange (see Fig. 1 and Fig. 3), with \vec{Z}_F perpendicular to the flange itself, and \vec{X}_F and \vec{Y}_F perpendicular and parallel to the rotation axes of the two engines, respectively. Let also $\mathcal{F}_C : \{O_C; \vec{X}_C; \vec{Y}_C; \vec{Z}_C\}$ be a frame attached to the gondola, with $\vec{Y}_C \parallel \vec{Y}_F$ and \vec{X}_C, \vec{Z}_C aligned with the gondola forward/upwards direction, respectively. Due to its mechanical design, the gondola cannot move along \vec{Y}_C (it is perpendicular to the rails) and the displacement between O_C and O_F along this direction is set to zero. We introduce the simplified situation depicted in Fig. 2, obtained by projecting the kinematics on the plane Σ spanned by $\{\vec{X}_F, \vec{Z}_F\}$. The projection of the rails on Σ consists of two segments, $\overline{AB}, \overline{CD}$ ¹, having length L and connected by a quarter of a circle \overline{BC} having radius λ . In order to simplify the following analysis, we place O_C at the center of \overline{BC} (see Fig. 2). We also define the points P_1 and P_2 as the

¹We use the symbol $\hat{}$ to denote a generic path

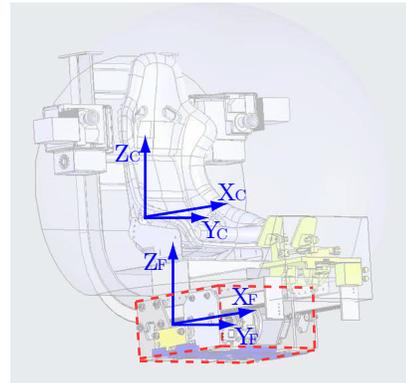


Fig. 1: Details of the novel actuated cabin for the CyberMotion Simulator. The dashed red box delimits the flange.

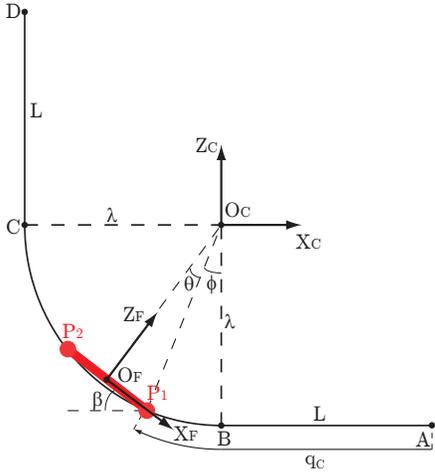


Fig. 2: Simplified sagittal view of the cabin, obtained by projection on the plane $\Sigma = \text{span}\{\vec{X}_F, \vec{Z}_F\}$

intersections of the rotational axes of the two engines with Σ . We will refer to these points as *pivots*. Since the motors are fixed on the rigid flange (the red thick segment in Fig. 2), by assuming negligible deformations, the distance between the two pivots stays constant, i.e.,

$$\|P_1 - P_2\| = d. \quad (2)$$

The mechanical design of the cabin also implies that P_1 and P_2 are bound to move on the path $\ell = \widehat{ABCD}$.

With these settings, we have that the relative orientation ${}^F R_C \in SO(3)$ from \mathcal{F}_C to \mathcal{F}_F can be parameterized by a single angle $\beta \in \mathbb{R}$ (see Fig. 2) as

$${}^F R_C = \begin{pmatrix} \cos \beta & 0 & -\sin \beta \\ 0 & 1 & 0 \\ \sin \beta & 0 & \cos \beta \end{pmatrix}. \quad (3)$$

We also have that the translation vector from O_F to O_C is in the form² ${}^F p_C = [{}^F p_{C,x} \ 0 \ {}^F p_{C,z}]^T \in \mathbb{R}^3$, with ${}^F p_{C,x}$ and ${}^F p_{C,z}$ functions of a certain configuration variable q_C . Since we chose O_F at the center of the flange, it follows that

$${}^F p_C = -{}^F R_C {}^C p_F, \quad (4)$$

where

$${}^C p_F = {}^C O_F = \frac{{}^C P_1 + {}^C P_2}{2}. \quad (5)$$

Finally, we select the *arc length* of the pivot P_1 along ℓ as the configuration variable q_C parameterizing the cabin status. We define q_C such that $q_C = q_{C_{min}} = 0$ when $P_1 \equiv A$ and $q_C = q_{C_{max}} = \lambda \frac{\pi}{2} + 2L - d$ when $P_2 \equiv D$. In Sect. III-A we will express ${}^F R_C$ and ${}^F p_C$ as functions of q_C .

A. Forward kinematics

By geometrical inspection of Fig. 2, it is possible to identify the following cases for the cabin forward kinematics:

²Superscripts are used to indicate the frames where quantities are expressed.

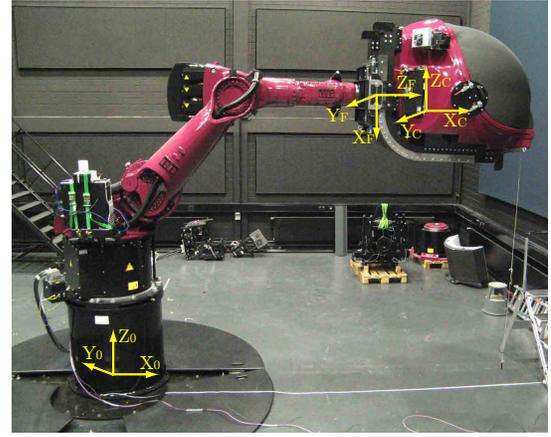


Fig. 3: Side view of CyberMotion Simulator with the actuated cabin

Case C₁: when P_1 and P_2 are both on a linear segment (\widehat{AB} or \widehat{CD}), the cabin joint reduces to a linear one. The frame \mathcal{F}_C can only translate with respect to \mathcal{F}_F , while the relative orientation is constant;

Case C₂: when the two pivots are on the arc \widehat{BC} the cabin joint reduces to a rotational one. Owing to the placement of the two frames of reference, the position of O_C is constant in \mathcal{F}_F and only the relative orientation of the two frames can vary;

Case C₃: when only one of the two pivots is on the curve \widehat{BC} , both the relative position and the relative orientation between the two frames can vary. The cabin joint behaves as a simultaneous linear and rotational joint.

The system can switch among these three cases only when one of the pivots is either on the point B or C . The corresponding *switching conditions* can be written in terms of q_C as

$$\begin{cases} P_2 \equiv B \Rightarrow q_C = q_{C_1} = L - d, & C_1 \Leftrightarrow C_3 \\ P_1 \equiv B \Rightarrow q_C = q_{C_2} = L, & C_3 \Leftrightarrow C_2 \\ P_2 \equiv C \Rightarrow q_C = q_{C_3} = L + \lambda(\frac{\pi}{2} - 2\theta), & C_2 \Leftrightarrow C_3 \\ P_1 \equiv C \Rightarrow q_C = q_{C_4} = L + \lambda\frac{\pi}{2}, & C_3 \Leftrightarrow C_1 \end{cases} \quad (6)$$

where $\theta = \arcsin(\frac{d}{2\lambda})$ (see Fig. 2), and $C_i \Leftrightarrow C_j$ indicates a possible switching between C_i and C_j . By construction it is $q_{C_{min}} \leq q_{C_1} \leq \dots \leq q_{C_4} \leq q_{C_{max}}$.

We can then address the three aforementioned cases

Case C₁: The cabin can be considered as a linear joint. When $P_1, P_2 \in \widehat{AB}$, i.e., $q_C \in [q_{C_{min}}, q_{C_1}]$, then it is $\beta = 0$, and ${}^F p_C = [q_C + \frac{d}{2} - L; 0; \lambda]$. Instead, when $P_1, P_2 \in \widehat{CD}$, i.e., $q_C \in [q_{C_4}, q_{C_{max}}]$, then it is $\beta = \frac{\pi}{2}$ and ${}^F p_C = [\frac{d}{2} - L + q_C - \lambda\frac{\pi}{2}; 0; \lambda]$.

TABLE II: Physical parameters of the cabin

L [m]	0.570
d [m]	0.200
λ [m]	0.504
θ [deg]	11.44

Case C₂: $P_1, P_2 \in \widehat{BC}$, i.e., $q_C \in [q_{C_2}, q_{C_3}]$. Owing to the choice of the frames, O_F is always located on a circumference centered in O_C and with \vec{Z}_F pointing towards it. By inspection, it follows that ${}^F p_C = [0; 0; -\lambda \cos \theta]$ and $\beta = \theta + \phi$, where $\phi = \frac{(q_C - L)}{\lambda}$ (see Fig. 2).

Case C₃: Here again, two different situations are possible. Consider first the one with $P_1 \in \widehat{AB}$, $P_2 \in \widehat{BC}$, i.e., $q_C \in [q_{C_1}, q_{C_2}]$. The constraint $P_1 \in \widehat{AB}$ implies that ${}^C P_1 = [L - q_C; 0; -\lambda]$. The position of the second pivot ${}^C P_2 = [{}^C P_{2,x}; 0; {}^C P_{2,z}]$ can be determined by imposing that $P_2 \in \widehat{BC}$ together with the constraint (2), i.e.,

$$\begin{cases} {}^C P_{2,x}^2 + {}^C P_{2,z}^2 = \lambda^2 \\ ({}^C P_{2,x} - {}^C P_{1,x})^2 + ({}^C P_{2,z} - {}^C P_{1,x})^2 = d^2 \end{cases} \quad (7)$$

With the parameters of Table II, this system always admits two solutions (intersections), and the correct one for this case corresponds to ${}^C P_{2,x} < 0$.

Consider now the second situation with $P_1 \in \widehat{BC}$, $P_2 \in \widehat{CD}$, i.e., $q_C \in [q_{C_3}, q_{C_4}]$. By imposing again the path constraint, it follows that ${}^C P_1 = [-\lambda \sin \phi; 0; -\lambda \cos \phi]$ and ${}^C P_2 = [-\lambda; 0; {}^C P_{2,z}]$. Similarly to the previous case, ${}^C P_{2,z}$ can be determined from the constraint (2), as being the positive solution of

$$(-\lambda + \lambda \sin \phi)^2 + ({}^C P_{2,z} + \lambda \cos \phi)^2 = d^2. \quad (8)$$

Finally, for both situations it is $\beta = \arctan \frac{{}^C P_{2,z} - {}^C P_{1,z}}{{}^C P_{1,x} - {}^C P_{2,x}}$ and the expression of ${}^F p_C$ and ${}^F R_C$ follows from (3–5).

The complete forward kinematics of the 7-DOF system, follows straightforwardly by plugging the cabin kinematics (3–5) from \mathcal{F}_F to \mathcal{F}_C into the manipulator kinematics from the world frame \mathcal{F}_0 to \mathcal{F}_F (see Fig. 3). For reasons related to the design of the motion cueing algorithm [12], [13], instead of using the Cartesian coordinates ${}^0 p_C = [x, y, z]^T$, we express the position of the cabin in \mathcal{F}_0 in cylindrical coordinates $\xi = [R \ \alpha \ z] \in \mathbb{R}^3$, defined as

$$\begin{cases} R &= \sqrt{x^2 + y^2} \\ \alpha &= \text{atan2}(y, x) \\ z &= z \end{cases},$$

As for the orientation of the cabin in \mathcal{F}_0 , we will use a different representation from what used in [12], [13]. In that case, with a fixed seat, a roll-pitch-yaw parametrization was used because the representation singularities were outside the workspace of the manipulator. Since, with the addition of the actuated cabin, these singular configurations are now within the workspace of the simulator, we resort to a unit Quaternion $Q = [\mu \ \epsilon^T]^T \in \mathbb{R}^4$ as orientation representation, where $\mu \in \mathbb{R}$ is the scalar part of the quaternion and $\epsilon \in \mathbb{R}^3$ is the vector part. Hereafter, for the sake of readability we will omit the dependency from $q \in \mathbb{R}^7$, unless strictly necessary.

B. Differential kinematics

The computation of the differential kinematics of the robot now requires to take into account the actuated cabin. First of all, we have to consider the contribution of the motion of the gondola w.r.t. the flange, i.e, how \dot{q}_C contributes to

the Cartesian/angular velocity of \mathcal{F}_C in \mathcal{F}_0 . This mapping is done represented by column vector $J_C(q) \in \mathbb{R}^{6 \times 1}$ defined as

$$J_C(q) = \begin{bmatrix} {}^0 R_F(q_M) & 0_{3 \times 3} \\ 0_{3 \times 3} & {}^0 R_F(q_M) \end{bmatrix} \begin{bmatrix} \frac{\partial {}^F p_C}{\partial q_C} \\ \begin{bmatrix} 0 \\ \frac{\partial \beta}{\partial q_C} \\ 0 \end{bmatrix} \end{bmatrix}, \quad (9)$$

where ${}^0 R_F(q_M) \in SO(3)$ is the rotation matrix from \mathcal{F}_F to \mathcal{F}_0 and the quantities ${}^F p_C$ and β were introduced in Sect. III. Subsequently, it is necessary to include the displacement form \mathcal{F}_F to \mathcal{F}_0 in the Jacobian matrix $J_M \in \mathbb{R}^{6 \times 6}$ mapping the joint velocity \dot{q}_M to the Cartesian/angular velocity of \mathcal{F}_F in \mathcal{F}_0 . Namely,

$$\bar{J}_M = \begin{bmatrix} I_{3 \times 3} & S(-{}^0 R_F {}^F p_C) \\ 0_{3 \times 3} & I_{3 \times 3} \end{bmatrix} J_M, \quad (10)$$

where $S(\cdot) \in \mathbb{R}^{3 \times 3}$ is the skew-symmetric matrix. Finally, the task Jacobian mapping the joint velocity $\dot{q} \in \mathbb{R}^7$ of the complete 7-DOF system in the cylindrical/angular velocity of the cabin in \mathcal{F}_0 is obtained by combining (10) and (9) as

$$J(q) = \begin{bmatrix} T(\xi(q)) & 0_{3 \times 3} \\ 0_{3 \times 3} & I_{3 \times 3} \end{bmatrix} [\bar{J}_M \ J_C], \quad (11)$$

where $T(\xi(q))$ is a suitable mapping matrix from Cartesian to cylindrical coordinates³. Note that the evaluation of J_q depends on the particular cases introduced in Sect. III-A.

Summarizing, we will take $r = [\xi^T \ Q^T] \in \mathbb{R}^7$ as *task variables* to be controlled with the Jacobian matrix $J(q)$ from (11).

IV. CONTROL LAW

We study now the problem of executing a reference task trajectory r_d , produced by the motion cueing algorithm introduced [12], [13]. By adopting a classic kinematic inversion formulation [21], the goal is to find a control law $u = f(q, r_d, r)$ that guarantees the execution of r_d . Moreover, since the motion cueing algorithm does not take explicitly into account the robot constraints of Table I, the reference trajectory can be not feasible. Ideally, whenever r_d is compatible with all the constraints, the robot should track it exactly. When one of the constraints is violated, the robot should try to replicate the reference trajectory as best as it can, i.e., by minimizing the norm of the Euclidean error $\|e(t)\| = \|r_d(t) - r(t)\|$. Formally, the joint limits and actuator constraints are

$$\begin{cases} \text{a)} & \forall i, \forall t \geq 0, \quad q_{i,\min} \leq q_i(t) \leq q_{i,\max} \\ \text{b)} & \forall i, \forall t \geq 0, \quad |\dot{q}_i(t)| \leq \dot{q}_{i,\max} \\ \text{c)} & \forall i, \forall t \geq 0, \quad |\ddot{q}_i(t)| \leq \ddot{q}_{i,\max} \end{cases}. \quad (12)$$

In addition to these limitations, the inversion scheme should avoid singularities or soften their effect by passing as ‘smoothly’ as possible through them.

³See [12] for more details.

In [12], [13], a complete solution able to deal online with these constraints was proposed and validated. Here we briefly summarize the main characteristics and omit a full technical description because of lack of space. The adopted inversion scheme was based on the use of Task Priority (TP) [22], by setting the cabin orientation as the highest priority task, and the cabin position as the lowest priority task. This approach was motivated by the fact that the motion cueing algorithm largely exploits the orientation of the gravitational acceleration to reproduce sustained cues. Therefore, close to singularities the realization of the correct orientation of gravity should be favored compared to the correct execution of the other task variables.

On top of this inversion scheme, an independent bang-bang control was used to deterministically avoid joint limits. This control becomes active at the very last moment when a joint can be stopped before hitting the limit, given the maximum velocities/accelerations characteristics of Table I. Once the bang-bang control is active for a joint, then that joint will be *locked* until it fully stops and a command from the TP inversion scheme tries to move it away from the limit. Note that this strategy guarantees safe behaviors in all conditions, but of course the motion performed can be heavily distorted as more joints become locked. In this respect, the addition of the new joint introduces a kinematic redundancy that can be exploited to soften these negative effects.

Therefore, we modify the TP control law of [12], [13] as

$$u = J_A^* w_A + J_{AB}^* (w_B - J_B J_A^* w_A) + (I - J_A^* J_A) (I - J_{AB}^* J_{AB}) w_C, \quad (13)$$

where w_i , $i = A, B, C$ are the different priority tasks with the priority decreasing from A to C , J_A and J_B are the corresponding subJacobians of J in 11 w.r.t. w_A and w_B , $J_{AB} = (J_B(I - J_A^* J_A))$, and the superscript $*$ denotes a matrix generalized inverse. Tasks $w_A \in \mathbb{R}^3$ and $w_B \in \mathbb{R}^3$ are selected as

$$w_A = \omega_d + K_O \Delta \epsilon \quad (14)$$

$$w_B = \dot{\xi}_d + K_P \Delta \xi \quad (15)$$

where $K_O, K_P \in \mathbb{R}^{3 \times 3}$ are positive definite diagonal gain matrices, $\Delta \xi = \xi_d - \xi$, and the orientation error is

$$\Delta \epsilon = \mu \epsilon_d - \mu_d \epsilon - S(\epsilon_d) \epsilon, \quad (16)$$

as illustrated in [23], [24].

The choice of vectors w_A and w_B follows the well-known CLIK paradigm [21] with the quaternion based implementation from [24] that ensures recovery of numerical drifts or tracking errors during motion. Moreover, to avoid ill-conditioning in the implementation of (13), we resorted to a singularity-robust pseudoinversion based on numerical filtering [25].

We can then exploit the redundancy by acting on the lowest priority task $w_C \in \mathbb{R}^7$. A common choice is to avoid singularities by maximizing the manipulability measure introduced by Yoshikawa [26]. In our case, we also

desire to avoid joint limits for the problems mentioned before. Therefore, we used the optimization criterion taken from [27],

$$H(q) = \left(1 - e^{-k \prod_{i=1}^n \frac{(q_i - q_{i,min})(q_{i,max} - q_i)}{(q_{i,max} - q_{i,min})^2}} \right) \sqrt{\det(J(q)J^T(q))} \quad (17)$$

where k is a design parameter and n is the number of joints. This function is a combination of Yoshikawa's manipulability measure with a penalty term that goes to zero close to joint limits. By maximizing $H(q)$ the robot will stay away from the singularities and joint limits. The threshold between these two actions is determined by k . The lowest priority task is therefore

$$w_C = k_C \left(\frac{\partial H(q)}{\partial q} \right), \quad (18)$$

where $k_C > 0$ is a user defined scalar that weights the task.

The joint velocity command produced by this scheme might not satisfy the actuators constraints in (12). In order to cope with these limitations, we applied the saturation scheme proposed in [12]. A uniform saturation of the velocity command u is used, when possible, so as to leave unaltered the direction of the task velocity. When no uniform scaling of u can solve the problem, a nonuniform saturation that does not change the direction of the acceleration \ddot{q} is adopted.

Because of the prioritization in (13), it would be desirable to adopt a saturation algorithm that takes into account the different priorities. A similar idea was developed in [28] by considering the case of the sole actuator velocity saturations, and by applying different uniform scalings to the tasks starting from the highest priority one until the lowest priority one. However, this approach cannot explicitly deal with acceleration constraints as in our case, and we also believe that a better result could be obtained by performing the prioritized scaling starting from the lowest priority task up to the highest one as opposite to [28]. This problem, however, falls outside the scope of the paper and will be a topic of future research.

V. SIMULATION RESULTS

In this Section we will analyze with an illustrative example the behavior of the combined CyberMotion Simulator/cabin with the proposed inverse kinematics and control law, and compare the results to the original system presented in [12], [13]. A video of this simulation is attached to the paper, while an experimental validation is under development.

In this example we want to simulate a constant linear deceleration ${}^C a_d = [-7 \ 0 \ 0]^T$ [m/s²] acting on the pilot, i.e., in \mathcal{F}_C . This deceleration is translated by the motion cueing algorithm into a desired task trajectory r_d . Namely, the cabin should be moved backwards (in \mathcal{F}_C) to reproduce the onset cue, and tilt downwards to reproduce the persistent cue by exploiting gravity. The manipulator starts from the configuration $q_M(t_0) = [0 \ -80 \ 60 \ 0 \ 20 \ 0]^T$ [deg] and the cabin from the configuration $q_C(t_0) = 1.34$ [m], both well inside the joint limits. The corresponding initial task

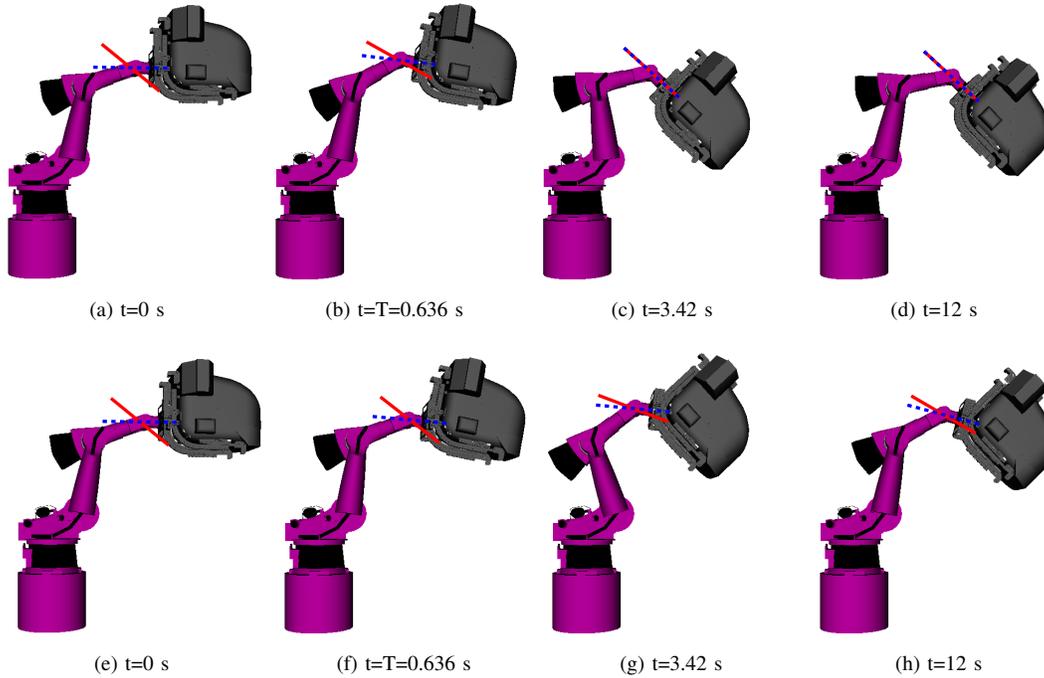


Fig. 4: Snapshots of the robot during the execution of the task. Figures 4(a-d) describe the motion with the cabin kept fixed as in [12], [13]. Figures 4(e-h) describe the motion by also exploiting the actuated cabin.

configuration is $r(t_0) = [2.55 \ 0 \ 3.63 \ 1 \ 0 \ 0 \ 0]^T$, with the cabin being perfectly vertical (i.e., $\vec{Z}_C = \vec{Z}_0$).

We chose this example because the sustained brake is a relevant cue for a car simulator. In [12], [13] the CyberMotion Simulator was used with a fixed seat to simulate a Formula 1 car. However, in that case the maximum simulated forward deceleration without significant false cues was up to $-4 \text{ [m/s}^2\text{]}$, while the simulated car was capable of braking well beyond this value. This simulation will then show that the new CyberMotion Simulator/cabin allows to increase the level of sustained deceleration reproducible without significant artifacts.

In the rest of this section, we will indicate with σ_P the smallest singular value of the primary task Jacobian J_A and with σ_S the smallest singular value of the secondary task ‘coupling’ matrix $J_B(I - J_A^*J_A)$. Furthermore, in the following plots we will represent the quantities relative to the fixed cabin case with solid lines, those relative to the actuated cabin with dashed lines, and any reference quantity with dotted lines.

The behavior of the robot during the task execution can be understood with the help of Fig. 4, that shows snapshots of the simulation for the non-actuated and actuated cabin case respectively. In the snapshots it is also depicted the current direction of the 5th link of the robot (blue dashed line) and its limit (solid red line). Consider first the fixed cabin case. From the starting position in Fig. 4(a), the cabin must move backwards along \vec{X}_C and at the same time must tilt downwards to correctly reorient the gravity vector. In order to reorient the seat without deviating from ξ_d , joints 3

and 5 are mainly exploited. In particular the latter is driven towards its limit and at the time $T = 0.636 \text{ [s]}$ the bang-bang controller activates and locks it (Fig. 4(b)). Hereafter, we will indicate this event in the plots with a vertical dashed line. At this point, the robot is not able to execute simultaneously Q_d and ξ_d anymore, joint 5 reaches its limit and the cabin visibly moves downwards (Fig. 4(c)). Finally, the onset cues expires and the desired orientation of the cabin is reached (Fig. 4(d)). The execution of the task trajectory is also shown in Fig. 5. In particular, starting at time T the robot is not able to track r_d anymore. While the primary task Q , despite a little deviation from the reference, follows Q_d (Fig. 5(b)), the execution of the secondary task ξ is visibly different from ξ_d (Fig. 5(a)). Since ξ_d is generated by the motion cueing algorithm to reproduce onset accelerations, this mismatch on ξ produces a very strong false cue during the transient. Note also that the desired forward deceleration $-7 \text{ [m/s}^2\text{]}$ is reached at about $t = 6 \text{ [s]}$.

Consider now the actuated cabin case. Starting from the same initial configuration (Fig. 4(e)), the reorientation is more efficiently executed by exploiting the 7th axis. At the time T (Fig. 4(f)) joint 5 is not locked and the robot can continue to track ξ_d to reproduce the onset cue (Fig. 4(g)). Eventually, the movement stops and the cabin remains in the desired orientation (Fig. 4(h)). The task trajectory is tracked accurately (see Fig. 5) and thus the desired deceleration is simulated without significant false cues (Fig. 6). The reference acceleration $-7 \text{ [m/s}^2\text{]}$ is also reached earlier, at about $t = 2.9 \text{ [s]}$.

The different performance in the two cases can be ana-

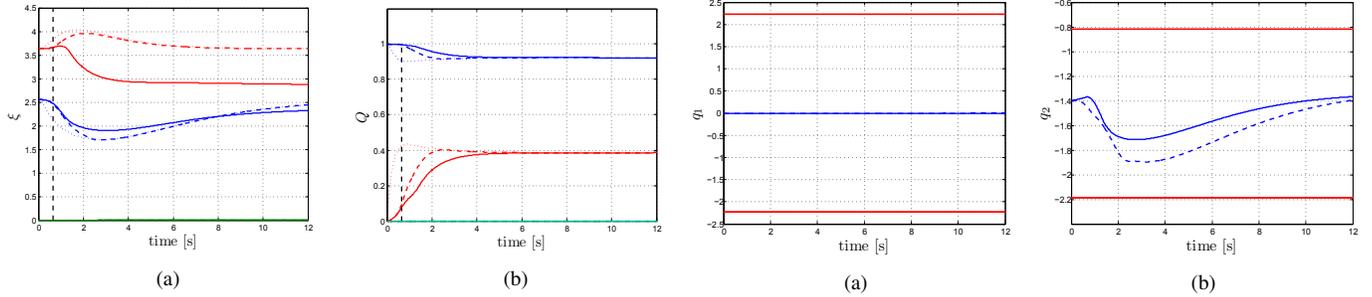


Fig. 5: Evolution of the task variables. Solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case and dotted lines represent the reference task. In Fig. 5(a), the color red is used for z , blue for R and green for α . In Fig. 5(b), the color blue is associated to μ , while green, red and cyan are used to depict the components of ϵ , in this exact order.

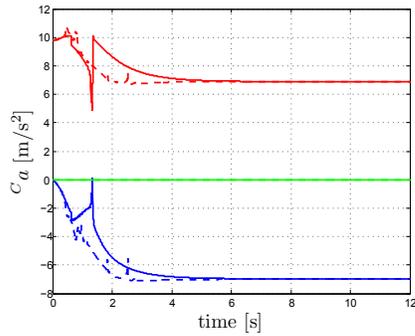


Fig. 6: Simulated acceleration C_a in \mathcal{F}_C . Solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case. Blue, green and red lines refer to accelerations along \vec{X}_C , \vec{Y}_C and \vec{Z}_C , respectively

lyzed by looking at the evolution of the joint variables during the task (Fig. 7). In the fixed cabin case, after being locked at the time T , joint 5 remains at its upper limit until the end (Fig. 7(e)). This also results in a larger evolution for joint 3 (Fig. 7(c)), that compensates for the loss of mobility. Clearly, at time T the modified manipulability function (17) rapidly goes to zero (Fig. 7(h)). Moreover, when joint 5 becomes locked, σ_S goes to zero as the subJacobian of J_B corresponding to the unlocked joints is singular (Fig. 8(b)). Anyway, thanks to the TP control, this singularity does not affect the primary task and σ_P is always greater than its starting value (Fig. 8(a)). With the addition of the actuated cabin, the burden on joints 3 and 5 is reduced and the vector q remains well inside the limits. Also, the manipulability function (17) never decreases below the starting value (Fig. 7(h)), and neither σ_P nor σ_S encounter singularities (Fig. 8).

VI. CONCLUSIONS

In this paper, we described the mechanical and control design of the new 7-DOF CyberMotion Simulator. In partic-

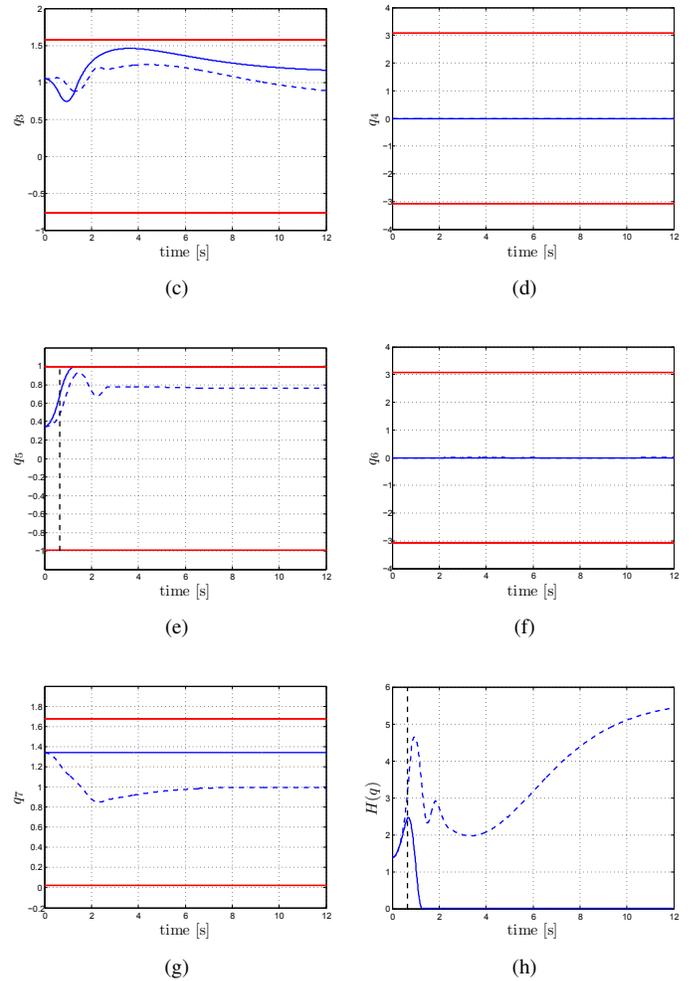


Fig. 7: Figures 7(a-g) show the evolution of the joint variables during the execution. The red lines indicate the limits for each joint. Fig. 7(h) shows the evolution of the modified manipulability function (17). In all the figures solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case.

ular, we focused on the actuation characteristics of the new cabin which behaves as a ‘hybrid’ joint: it can switch from being a linear/rotational one during its operation. We then addressed the issue of exploiting the actuation redundancy to improve the behavior of the system when reproducing

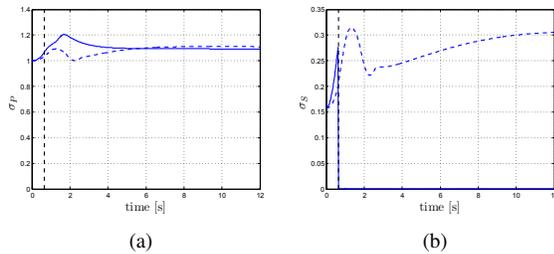


Fig. 8: Evolution of σ_P and σ_S during the execution. Solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case.

acceleration profiles typical of race cars. Simulation results confirmed the validity of the modeling/control approach.

In the future, we plan to use the new CyberMotion Simulator as a versatile tool for reproducing the behavior of different kind of vehicles, ranging from cars, airplanes and helicopters. In this sense, a live demo of the system was already presented at the ILA Air Show in Berlin (June 2010) [20] where the robot was used as an helicopter simulator. We are also working on the control side of the system, trying to find suitable optimization criteria (also depending on the particular simulated vehicle) to further improve the behavior of the system.

ACKNOWLEDGMENTS

The authors wish to thank Michael Kerger and Dr. Harald Teufel for their intensive support. They also gratefully acknowledge the support of the Max Planck Society and the WCU (World Class University) program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (R31-2008-000-10008-0).

REFERENCES

- [1] L. J. Hettinger and M. W. Haas, *Virtual and Adaptive Environments: Applications, Implications, and Human Performance Issues*. LEA, 2009.
- [2] P. F. Hokayem and M. W. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.
- [3] A. Bicchi, M. Buss, M. O. Ernst, and A. Peer, *The Sense of Touch and Its Rendering: Progress in Haptics Research*. Springer Tracts in Advanced Robotics, 2008.
- [4] J. Burki-Cohen, T. H. Go, and T. Longridge, "Flight simulator fidelity considerations for total air line pilot training and evaluation," *Proc. of the AIAA Modeling and Simulation Technologies Conference*, 2001.
- [5] Robotic Vision - To 2020 and Beyond, *The Strategic Research Agenda for Robotics in Europe*. EUROP, 2009.
- [6] J. Heindl, M. Otter, H. Hirschmüller, M. Frommberger, N. Sporer, F. Siegert, and H. Heinrich, "The robocoaster as simulation platform – Experiences from the "first authentic mars flight simulation"," *Proc. of the 1st Motion Simulator Conference*, 2005.
- [7] T. Bellmann, M. Otter, J. Heindl, and G. Hirzinger, "Real-time path planning for an interactive and industrial robot-based motion simulator," *Proc. of the 2nd Motion Simulator Conference*, 2007.
- [8] H. J. Teufel, H.-G. Nusseck, K. A. Beykirch, J. S. Butler, M. Kerger, and H. H. Bühlhoff, "MPI motion simulator: Development and analysis of a novel motion simulator," *Proc. AIAA Modeling and Simulation Technologies Conference*, 2007.

- [9] H. H. Bühlhoff, "The MPI motion simulator: A new approach to motion simulation with an anthropomorphic robot arm," *Proc. of the 2nd Motion Simulator Conference*, 2007.
- [10] L. Pollini, M. Innocenti, and A. Petrone, "Novel motion platform for flight simulators using an anthropomorphic robot," *J. of Aerospace Computing, Information, and Communication*, vol. 5, pp. 175–196, 2008.
- [11] Robocoaster, "www.robocoaster.com."
- [12] P. Robuffo Giordano, C. Masone, J. Tesch, M. Breidt, L. Pollini, and H. H. Bühlhoff, "A novel framework for closed-loop robotic motion simulation - Part I: Inverse kinematics design," *Proc. of the 2010 IEEE Int. Conf. on Robotics and Automation*, pp. 3876–3883, 2010.
- [13] —, "A novel framework for closed-loop robotic motion simulation - Part II: Motion cueing design and experimental validation," *Proc. of the 2010 IEEE Int. Conf. on Robotics and Automation*, pp. 3896–3903, 2010.
- [14] Desdemona: the next generation in movement simulation. [Online]. Available: www.tno.nl
- [15] VTI simulator III. [Online]. Available: www.vti.se
- [16] E. L. Groen and W. Bles, "How to use body tilt for the simulation of linear self motion," *J. of Vestibular Research*, vol. 14, no. 5, pp. 375–385, 2004.
- [17] M. A. Nahon and L. D. Reid, "Simulator motion-drive algorithms: A designer's perspective," *J. of Guidance, Control, and Dynamics*, vol. 13, no. 2, pp. 356–362, 1990.
- [18] P. R. Grant and L. D. Reid, "Motion washout filter tuning: Rules and requirements," *J. of Aircraft*, vol. 34, no. 2, pp. 145–151, 1997.
- [19] D. R. Berger, J. Schulte-Pelkum, and H. H. Bühlhoff, "Simulating believable forward accelerations on a Stewart motion platform," *ACM Transactions on Applied Perception*, vol. 7, no. 1–5, pp. 1–27, 2010.
- [20] ILA Berlin Air Show 2010. [Online]. Available: http://www.ila-berlin.de
- [21] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy," *The International Journal of Robotics Research*, vol. 10, pp. 410–425, 1991.
- [22] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions On Robotics And Automation*, vol. 13, no. 3, pp. 398–410, 1997.
- [23] B. Siciliano, "On the use of quaternions for robot interaction control tasks," *Proceedings of the 5th Symposium on Methods and Models in Automation and Robotics*, pp. 739–746, 1998.
- [24] F. Caccavale and B. Siciliano, "Quaternion-based kinematic control of redundant spacecraft/manipulator systems," *Proceedings of the 2001 IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 435–440, 2001.
- [25] A. A. Maciejewski and C. A. Klein, "Numerical filtering for the operation of robotic manipulators through kinematically singular configurations," *J. of Robotic Systems*, vol. 5, no. 6, pp. 527–552, 1985.
- [26] T. Yoshikawa, "Manipulability of robotic mechanisms," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.
- [27] B. J. Nelson and P. K. Khosla, "Strategies for increasing the tracking of an eye-in-hand system by singularity and joint limit avoidance," *The International Journal of Robotics Research*, vol. 14, no. 3, pp. 255–269, 1995.
- [28] F. Arrichiello, S. Chiaverini, G. Indiveri, and P. Pedone, "The null-space based behavioral control for a team of cooperative mobile robots with actuator saturations," *The 2009 IEEE/RSJ Int. Conf on Intelligent Robots and Systems*, pp. 5911–5916, 2009.