robotics

MDPI

Article

Null-Space Minimization of Center of Gravity Displacement of a Redundant Aerial Manipulator

Yash Vyas †, Alberto Pasetto †, Victor Ayala-Alfaro, Nicola Massella and Silvio Cocuzza *

Department of Industrial Engineering, University of Padova, 35131 Padova, Italy
* Correspondence: silvio.cocuzza@unipd.it
† These authors contributed equally to this work.

Abstract: Displacements of the base during trajectory tracking are a common issue in the control of aerial manipulators. These are caused by reaction torques transferred to the base due to the manipulator motion and, in particular, to the motion of its center of gravity. We present a novel approach to reduce base displacements of a kinematically redundant aerial manipulator by using null-space projection in the inverse kinematic control. A secondary objective function minimizes the horizontal displacement of the manipulator center of gravity. We test this algorithm on different trajectories for both three and four degrees of freedom (DOF) manipulators in a simulation environment. The results comparing our algorithm with inverse kinematic control without the null-space projection show up to an 80% reduction in the end-effector position error and an average of about 56% reduction in maximum base displacement. The simulation implementation also runs faster than in real-time in our code implementation. We provide a workspace analysis based on multiple stopping criteria such as excessive base displacement, joint velocities and end-effector position error for the 3 and 4 DOF manipulators. As expected, the 4 DOF manipulator has a larger workspace.

Keywords: aerial manipulation; UAV; robot; kinematic control; redundancy; dynamic balancing

Citation: Vyas, Y.; Pasetto, A.; Ayala-Alfaro, V.; Massella, N.; Cocuzza, S. Null-Space Minimization of Center of Gravity Displacement of a Redundant Aerial Manipulator. *Robotics* **2023**, *12*, 31. https://doi.org/10.3390/robotics12020031

Academic Editor: Roberto Sabatini

Received: 7 December 2022
Revised: 31 January 2023
Accepted: 13 February 2023
Published: 21 February 2023

1. Introduction

Aerial manipulation is an active field of research for industrial and service application scenarios. The use of these robotic systems for inspection, maintenance and assembly of structures offers benefits over the human operation, particularly in safety-critical applications. They enable precision, maneuverability, and dexterity, and can be fitted to conduct many different types of tasks, such as applying a force, gripping and moving objects or non-contact sensing.

An aerial manipulator consists of an unmanned aerial vehicle (UAV) base and a robotic manipulator mounted on it. The most typical base is that of a multicopter with several rotors that can hover, which is required to keep the base stable while the manipulator carries out its task [1]. The type of manipulator depends on the task to be carried out. For a complex task involving interaction with objects, a multiple serial link arm is a commonly utilized option [2–4]. Various designs for these systems have been extensively tested such as [5–8].

Aerial manipulation control methods vary from decoupled to fully coupled, based on the coupling of the model and control of the UAV and manipulator [1]. The control of these aerial manipulators is an open research problem, as the base is not fixed and maintains its elevation and attitude using thrust from the rotors. This means that during a manipulation task, the base is prone to displacement caused by the reaction forces and torques generated from the manipulator motions. This can cause instability in the UAV and imprecision during the manipulation task.

The control of floating base manipulators has been addressed in space robotics, where we can assume conservation of momentum [9–11]. Some inverse kinematic control algorithms such as the ones based on the Generalized Jacobian [12] account for the base motions

caused by the manipulator, and an adaptation of the Generalized Jacobian for aerial manipulation is presented in [13]. Other methods address the issue at a mechanical design level, through counter-balancing arms or masses [14,15]. There are various methods of formulating the inverse kinematics of a redundant space manipulator as a linear algebraic problem to be solved at each time step [10,16].

If we consider only the manipulator kinematics, we can define an inverse kinematic approach that conducts multi-objective optimization with the secondary task of minimizing reaction torques during manipulation. Some approaches to reducing reaction torques for space manipulation have been researched in [10,16,17]. Exploiting null-space projection to improve the task execution precision of an aerial manipulator has been attempted by Ivanovic et al. [18], where the end-effector pose obtained from simulating the UAV motion is corrected to match the desired end-effector pose obtained in trajectory planning through the null space of the manipulator.

Our approach in this research paper focuses on the inverse kinematic control of a kinematically redundant aerial manipulator, where the UAV control is decoupled from the manipulator control. While some prior research has explored the control of redundant manipulators [19], there has been limited exploration into reducing the reaction torques exerted by the manipulator on the base in this scenario. We use null-space gradient projection to satisfy an additional secondary task that minimizes horizontal displacement of the manipulator center of gravity (CoG), where the primary task is to track a trajectory with the end-effector. By doing this, we also reduce the reaction torque on the base, and as a result, we reduce the translations of the base.

We model the following case: the base of the aerial manipulator is a UAV multicopter, and the manipulator is a kinematically redundant serial link arm with three or four degrees of freedom (DOF) actuated in a 2D plane. The manipulation task chosen is to track three different types of trajectories with the end-effector in the 2D plane. The UAV controller for this task is therefore simplified, which allows us to better understand the impact of manipulator motions on the reaction torques and base displacement.

The following section (Section 2) provides the background on how we model the dynamics and control of the UAV, the kinematic and dynamic modeling of a floating base manipulator, and a general introduction to inverse kinematic control of redundant manipulators.

Section 3 explains our algorithm for the minimization of the manipulator CoG displacement using an objective function and null-space projection for the inverse kinematic control and simulation of the dynamics of the system.

Section 4 outlines our simulation environment, task trajectories, and results from the simulations. We compare both 3 and 4 DOF manipulators with the simple pseudoinverse kinematic control and our approach in terms of base displacements, reaction torques, and end-effector precision.

Section 5 analyzes the workspace of the 3 and 4 DOF manipulators with the developed algorithm. The Conclusion (Section 6) summarizes the results and the possible future work related to this research.

2. System Modelling and Control

2.1. Unmanned Aerial Vehicle Base

The base which moves in 3D space with the mounted manipulator is modeled as a multicopter with 4 or more uniaxial rotors, i.e., a Hexacopter or Octocopter which can maintain a hover state. The parameters for the modeled UAV are based on the DJI S1000 Octocopter (Shenzhen, China); however, the developed inverse kinematic control algorithm can be used for any kind of UAV base where the control forces applied on the base are known. The first joint of the serial manipulator is mounted directly below the UAV base CoG. We use a simplified control model based on applied thrust and attitude torques for the UAV, assuming that a low-level controller exists that sets the rotor speeds to achieve these high-level control inputs.

We define the following reference frames: the inertial frame is denoted with subscript I, while the body-fixed frame about the base CoG is denoted with subscript B. The direction parallel to the thrust force is set as the body fixed z_B axis, which is parallel to the inertial z_I axis when the UAV is at hover state, i.e., the control forces maintain close to perfect equilibrium conditions. The UAV y_B axis is aligned to be parallel to the joint axes of the planar serial manipulator joints. These are illustrated in Figure 1.

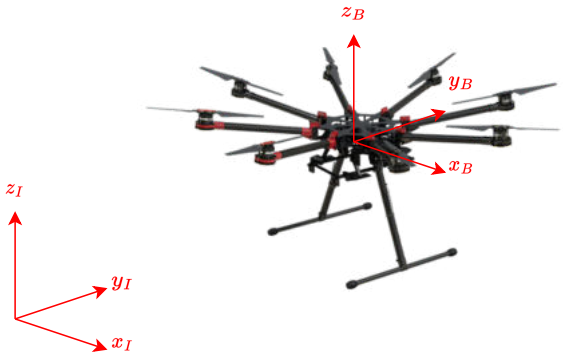

Figure 1. Reference frames for our UAV model (DJI S1000). The inertial frame is denoted with I and UAV base-fixed frame with B.

In our research, we model a serial manipulator that moves on the xz plane using joints whose rotation axes are parallel to y_B. Thus, the system is simplified into a 2D planar system, where the UAV can move vertically upwards on the z_B axis due to the thrust force U_1, and rotate about its CoG centered y_B axis through roll torque U_2. We represent the planar UAV with the following generalized coordinates: x which is the x-axis position of B in the inertial frame I, z which is the z-axis position of B in frame I, and ϕ which is the roll angle of UAV (angle between z_B and the CoG centered axis parallel to z_I).

The dynamics of the UAV in the xz plane is modeled with the following equations, where m_B is the UAV mass and I_B is the rotational inertia about y_B [13]:

$$m_B\ddot{x} = -U_1 \sin(\phi) \tag{1}$$

$$m_B\ddot{z} = U_1 \cos(\phi) - m_B g \tag{2}$$

$$I_B\ddot{\phi} = U_2 \tag{3}$$

The control algorithm for the aerial manipulator is decoupled, with an independent Proportional-Integral-Derivative (PID) controller for inputs U_1 and U_2 that correct for error from hover conditions ($\ddot{z} = \dot{z} = z = 0, \ddot{\phi} = \dot{\phi} = \phi = 0$). It is important to note from (1) that for a uni-directional arrangement of the rotors such as in our system, the x displacement cannot be independently controlled and is usually corrected through a combination of U_1 and U_2 thanks to an outer control loop which is not modeled in this work. The PID control input is set according to the following equations:

$$U_1 = -k_{P_z} z - k_{D_z} \dot{z} - k_{I_z} \int_0^t z dt \tag{4}$$

$$U_2 = -k_{P_\phi} \phi - k_{D_\phi} \dot{\phi} - k_{I_\phi} \int_0^t \phi dt \tag{5}$$

where $k_{P\_}$, $k_{D\_}$ and $k_{I\_}$ are the proportional, derivative and integral coefficients for the controller for subscript coordinates z and ϕ. The geometric parameters and forces on the UAV are illustrated in Figure 2. We note that aerial manipulator motion causes a reaction torque on the mount point of the robot arm. This reaction torque causes rotation and displacement of the UAV, which are counteracted by our control inputs U_1 and U_2. Therefore, minimizing this reaction torque will improve the stability and precision of our aerial manipulator.

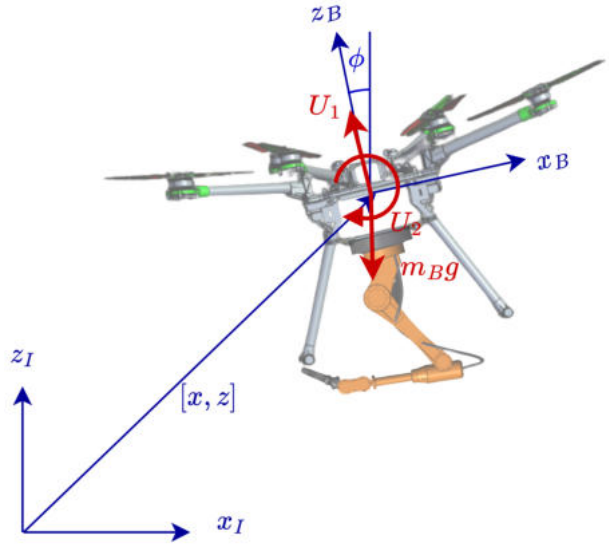

Figure 2. Coordinates, reference frames and control forces applied on the UAV. Blue shows the reference frames and coordinate parameters x, z and ϕ, while red shows the control inputs and gravity force.

2.2. Floating-Base Manipulator

In order to design an inverse kinematic controller for the aerial manipulator, it is necessary to recall the kinematics and dynamics of the coupled base-manipulator system. We model the system as a floating base manipulator under the influence of external forces and torques, using the conventions commonly used in space manipulator mechanics [20].

Through differential kinematics, the end-effector velocity $\boldsymbol{v}_e \in \mathbb{R}^6$ can be expressed as a function of joint velocities ($\dot{\boldsymbol{q}}$) and base velocities ($\dot{\boldsymbol{x}}_b \in \mathbb{R}^6$) by defining the manipulator Jacobian $\boldsymbol{J} \in \mathbb{R}^{6\times n}$, and the base Jacobian $\boldsymbol{J}_b \in \mathbb{R}^{6\times 6}$:

$$\boldsymbol{v}_e = \boldsymbol{J}\dot{\boldsymbol{q}} + \boldsymbol{J}_b\dot{\boldsymbol{x}}_b \tag{6}$$

The forward dynamics, accounting for forces and torques at the base and end-effector, are modeled by the following equation [21], omitting forces acting on the end-effector which are null for trajectory tracking tasks:

$$\begin{bmatrix} \boldsymbol{H}_b & \boldsymbol{H}_{bm} \\ \boldsymbol{H}_{bm}^T & \boldsymbol{H}_m \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{x}}_b \\ \ddot{\boldsymbol{q}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{c}_b \\ \boldsymbol{c}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_b \\ \boldsymbol{\tau} \end{bmatrix} \tag{7}$$

where:

- $\ddot{\boldsymbol{x}}_b \in \mathbb{R}^6$ is the vector of linear and angular accelerations of the base with respect to the inertial frame;
- $\boldsymbol{c}_b \in \mathbb{R}^6$ and $\boldsymbol{c}_m \in \mathbb{R}^n$ are non-linear centrifugal and Coriolis terms as a function of the base velocities $\dot{\boldsymbol{x}}_b$, joint angles $\boldsymbol{q}$ and joint velocities $\dot{\boldsymbol{q}}$;
- $\boldsymbol{\tau} \in \mathbb{R}^n$ are the joint torques;
- $\boldsymbol{F}_b$ is the vector of generalized forces and torques acting on the base, consisting of the control inputs and gravity force and torque ((1)–(3));
- $\boldsymbol{H}_b \in \mathbb{R}^{6\times 6}$ is the base inertia matrix [13,20];
- $\boldsymbol{H}_m \in \mathbb{R}^{n\times n}$ is the manipulator inertia matrix [13,20];
- $\boldsymbol{H}_{bm} \in \mathbb{R}^{6\times n}$ is the base-manipulator coupling inertia matrix [13,20].

2.3. Inverse Kinematic Control

If we neglect the base motion in (6), the equation can be re-arranged to solve for the joint velocities that allow achieving the desired end-effector velocity $\boldsymbol{v}_e$:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\boldsymbol{v}_e \tag{8}$$

However, for a direct inversion, the degrees of freedom of the manipulator n and the dimension of the task $\boldsymbol{v}_e$ have to be the same, so that the Jacobian $\boldsymbol{J}$ is a square matrix. For a 2D planar trajectory tracking task with joints actuated in the task plane, $\boldsymbol{v}_e \in \mathbb{R}^2$, so any manipulator with more than 2 joints is kinematically redundant if the orientation of the end-effector is not considered. This means that there are infinite solutions of $\dot{\boldsymbol{q}}$ that satisfy the task.

Two general approaches can be followed to find a unique solution for the joint velocities in the case of a redundant manipulator: (1) add additional tasks along with the velocity tracking to extend the Jacobian [9] or (2) minimize the joint velocities in the task-space. The second option can be carried out by finding the Moore–Penrose pseudoinverse $\boldsymbol{J}^+$, which finds the minimized solution for $\dot{\boldsymbol{q}}$ assuming equal weight for all joint velocities, while tracking the desired end-effector velocity (6):

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1}\boldsymbol{v}_e = \boldsymbol{J}^+\boldsymbol{v}_e \tag{9}$$

An extension of this approach allows the addition of a secondary objective through the projection of an arbitrary vector $\dot{\boldsymbol{q}}_0$ in the null space of $\boldsymbol{J}$, known as the gradient projection method [9]:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^+\boldsymbol{v}_e + (\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\dot{\boldsymbol{q}}_0 \tag{10}$$

3. Null-Space Gradient Projection Method to Minimize Base Displacement

3.1. Minimizing the Manipulator Horizontal Center of Gravity Displacement

We utilize the gradient projection method to minimize the displacement of the CoG of the manipulator with respect to the UAV CoG (10). According to [22], to optimize a scalar objective function of the joint variables $w(\boldsymbol{q})$ locally, we can define $\dot{\boldsymbol{q}}_0$ as:

$$\dot{\boldsymbol{q}}_0 = k_0\left(\frac{\partial w(\boldsymbol{q})}{\partial \boldsymbol{q}}\right)^T \tag{11}$$

where $k_0 \in \mathbb{R}^+$ is a weighting coefficient. Some typical definitions of $w(\boldsymbol{q})$ aim to maximize the distance from joint limits, the distance between the manipulator and obstacles, or the manipulability measure.

As we want to reduce the horizontal translation of the UAV caused by the manipulator motion, we attempt to minimize the UAV roll caused by the reaction torque exerted by the manipulator on the UAV. The gravity torque related to the displacement of the manipulator CoG from the base CoG represents the main contribution of this torque [13]. So, we formulate an objective function $w(\boldsymbol{q})$ that minimizes this displacement as a function of joint angles.

In the 2D planar case, the gravity torque on the manipulator about the UAV CoG can be calculated as:

$$\tau_g = x_{G,man}(\boldsymbol{q}) \cdot m_{man}g \tag{12}$$

where $x_{G,man}$ is the horizontal coordinate of the manipulator center of gravity with respect to the UAV body frame, and m_{man} is the manipulator's total mass, which is constant. Therefore, we define the objective function that has to be maximized as:

$$w = -(x_{G,man}(\boldsymbol{q}))^2 \tag{13}$$

which presents a local maximum when τ_g is null. For a planar serial manipulator with n DOF and with the first joint vertically aligned with the base CoG, $x_{G,man}$ and the objective function can be calculated from joint angles and manipulator geometry through:

$$x_{G,man}(\boldsymbol{q}) = \frac{\sum_{i=1}^{n} m_i \cdot \left(\sum_{j=1}^{i-1} l_j \cdot \sin(\theta_j) + a_i \cdot \sin(\theta_i)\right)}{\sum_{i=1}^{n} m_i} \tag{14}$$

where m_i and l_i are, respectively, the mass and the length of the ith link, a_i is the distance from ith joint to ith link CoG, $\theta_j = \sum_{k=1}^{k=j} q_k$ is the angle between the jth link and the base z axis, and $\theta_i = \sum_{k=1}^{k=i} q_k$ is the angle between the ith link and the base z axis.

However, our inverse kinematic equation (10) requires the derivative of our objective function $\frac{\partial w(\boldsymbol{q})}{\partial \boldsymbol{q}}$ and hence of (14). To simplify this computation we derive a linear approximation $\boldsymbol{d}_q \in \mathbb{R}^n$, $\boldsymbol{d}_q \approx \frac{\partial w(\boldsymbol{q})}{\partial \boldsymbol{q}}$, around the current state $\boldsymbol{q} = \boldsymbol{q}_k$ where k is the current discrete time step. This is found through numerical calculation of the change in $w(\boldsymbol{q})$ from perturbing each coordinate q_i in q for variable $i = \{1, 2, ..., n\}$ with a small ϵ and derivation of the corresponding entry of row i in $\boldsymbol{d}_q$, d_{q_i}:

$$d_{q_i} = \frac{w(\boldsymbol{q}_k + \boldsymbol{\epsilon}) - w(\boldsymbol{q}_k)}{\epsilon} \tag{15}$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^n$ is the perturbation vector, that is $\epsilon_j = \epsilon$ for row $j = i$ and $\epsilon_j = 0$ for $j \neq i$.

We note that when solving the inverse kinematics through (10) and (11), the choice of the parameter k_0 in (11) influences the optimization of the objective function, as it is related to the importance given to the secondary objective with respect to the minimum joint velocities solution (obtained for $k = 0$). The optimal value can differ based on the DOF of the manipulator and the type of trajectory, and the relative importance of minimizing joint velocities (which are closely related to energy consumption).

3.2. Inverse Kinematics and Simulation Algorithm

The kinematic control algorithm proposed in Section 3.1 is implemented in a MATLAB simulation environment that recursively solves the kinematics and dynamics of free-base robotic systems [16]. In particular, we calculate the joint velocities that allow tracking the desired end-effector trajectory at each timestep.

Since the base of the manipulator is not fixed, we also need to compute the movements of the base and their effects on the end-effector velocity. In order to achieve this, we solve the inverse kinematics in the body frame by considering the manipulator mounted on a fixed base, and then we calculate the motion of the UAV by solving the dynamics of the system at each timestep. In particular, for the determination of UAV acceleration, we exploit the formulation of the equations of motion (7).

We rearrange (7) to solve for $\ddot{\boldsymbol{x}}_b$; in our case, joint accelerations are known since joint velocities are controlled, so the second row is not considered:

$$\ddot{\boldsymbol{x}}_b = \boldsymbol{H}_b^{-1}(\boldsymbol{F}_b - \boldsymbol{c}_b - \boldsymbol{H}_{bm}\ddot{\boldsymbol{q}}). \tag{16}$$

By discretizing and numerically integrating the base accelerations (16), we can simulate the UAV motion and calculate the desired velocity of the end-effector (relative to the base) to be used in the inverse kinematics.

We assume that joint angles and UAM configuration do not change significantly between two subsequent timesteps. Therefore, at each timestep k the following procedure is used:

1. The current value of the objective function is calculated using the joint angles of previous timestep through (10) and $\dot{\boldsymbol{q}}_{0,k}$ is calculated using (15) and (11).

2. Since the desired trajectory is defined in the inertial frame, we calculate the end-effector velocity in the base frame Σ_B through the following relative velocity equation, providing the frame dependency explicitly as the left superscript:

$$^{B}\boldsymbol{v}_{e,k} = {}^{I}\boldsymbol{v}_{e,k} - {}^{I}\boldsymbol{v}_{b,k-1} - {}^{I}\boldsymbol{\omega}_{b,k-1} \times {}^{I}\boldsymbol{r}_{b,EE,k-1} \tag{17}$$

where $\boldsymbol{v}_{b,k-1}$ and $\boldsymbol{\omega}_{b,k-1}$ are, respectively, the linear and angular velocity of the base and $\boldsymbol{r}_{b,EE}$ is the vector from the base CoG to the end-effector; $k-1$ subscript indicates that values are taken from the previous timestep.

3. The inverse kinematics is solved through (10) where the end-effector desired velocity is substituted with (17) and the manipulator Jacobian $\boldsymbol{J}$ is calculated from joint angles at the previous timestep. We omit the dependency of the Jacobian on joint angles and add the subscript $k-1$ indicating that the previous timestep is used in the implementation of the inverse kinematic control equation:

$$\dot{\boldsymbol{q}}_k = \boldsymbol{J}_{k-1}^{+} \, {}^{B}\boldsymbol{v}_{e,k} + (\boldsymbol{I} - \boldsymbol{J}_{k-1}^{+}\boldsymbol{J}_{k-1})\dot{\boldsymbol{q}}_{0,k-1} \tag{18}$$

Joint accelerations are computed through the numerical differentiation of joint velocities and joint angles are computed by the numerical integration of joint velocities.

4. In order to solve the dynamics of the system and compare the base accelerations, we need to calculate all the terms of the right side of (16) and, in particular, the vector $\boldsymbol{F}_{b,k}$, which contains all the external forces (rows 1–3) and torques (rows 4–6) on the base. From the state of the UAM at the previous timestep, we calculate the control thrust $U_{1,k-1}$ (4), the control torque $U_{2,k-1}$ (5), the gravity force, and the gravity torque acting on the base. Then we combine all these components in order to obtain $\boldsymbol{F}_{b,k}$; in our planar case:

$$\boldsymbol{F}_{b,k} = \begin{bmatrix} -U_{1,k-1} \cdot sin(\phi_{k-1}) \\ 0 \\ U_{1,k-1} \cdot cos(\phi_{k-1}) \\ 0 \\ U_{2,k-1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -m_T g \\ \boldsymbol{r}_{G,man}(\boldsymbol{q}_k) \times \begin{bmatrix} 0 \\ 0 \\ -m_{man} g \end{bmatrix} \end{bmatrix} \tag{19}$$

where $\boldsymbol{r}_{G,man}(\boldsymbol{q}_k)$ is the vector from the base CoG to the manipulator CoG, m_T is the total mass of the UAM, and m_{man} is the mass of the manipulator.
The matrices $\boldsymbol{H}_m$, $\boldsymbol{H}_b$ and $\boldsymbol{H}_{bm}$ are computed from the joint angles, base position and base orientation, for which the full derivation is presented in [20].

5. Through (16), the dynamics is solved and the base accelerations are computed:

$$\ddot{\boldsymbol{x}}_{b,k} = \boldsymbol{H}_b^{-1}(\boldsymbol{F}_{b,k} - \boldsymbol{c}_{b,k} - \boldsymbol{H}_{bm}\ddot{\boldsymbol{q}}_k). \tag{20}$$

Base velocities and positions are calculated by numerically integrating (16).

This procedure was implemented in the above mentioned MATLAB simulation environment.

4. Simulation Environment and Results

4.1. Simulation Environment Setup

As described in Section 3.2, the UAM kinematic control algorithm is implemented in a MATLAB simulation environment that recursively solves the kinematics and dynamics of free-base robotic systems [16]. This simulator was originally developed for space robotic systems but has been adapted to take into consideration the effect of the gravity force, as well as external forces, such as the control forces of the UAV platform. For all the simulations presented in this paper, a discretization timestep of 2×10^{-3} s is used and the weighting coefficient k_0 is set to 2×10^3.

We consider two different kinematically redundant arm configurations, 3 DOF and 4 DOF, both composed of identical links with the same geometric, mass, and inertial properties. Both configurations are tested using the same set of trajectories to be tracked by the end-effector. For each trajectory, the initial joint angles are set so that the base and manipulator CoGs are aligned and the manipulator is in a non-singular configuration. At the beginning of each simulation, the whole system is in equilibrium (i.e., the UAM is in a hovering state). Moreover, the starting point of the end-effector is set at the beginning of each trajectory. To account for the difference in length between the 3 and 4 DOF arms a different starting position of the end-effector along the z_I axis is considered.

The manipulator arm links are modeled to also account for the joint actuator weights, and hence their mass is not uniformly distributed. A point mass models the actuator mass (equal for the three joints), resulting in the link CoG being biased towards one of the ends of the link. A visual representation of this distribution is depicted in Figure 3, with the masses of the link and actuator motor and combined CoG shown. The last link has the same mass distribution since it is assumed that the end-effector weight is equal to one of the motors.

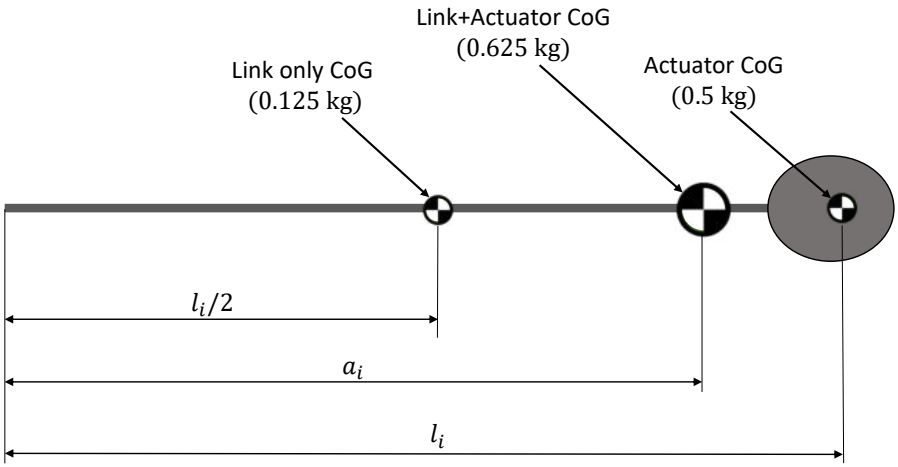

Figure 3. Mass distribution and CoG of the manipulator links.

The UAV and manipulator description, including kinematic and dynamic parameters, is outlined in Table 1. The UAV parameters are the ones of the high payload octocopter DJI S1000 available in our lab (as shown in Figure 1). The manipulator parameters are based on a preliminary design and on reasonable weights for links and motors based on commercial parts commonly available. Regardless, our algorithm is also applicable to different manipulator designs with different link lengths, masses, and mass distributions.

Table 1. Kinematic and dynamic UAM model parameters used in simulation experiments.

Parameter	Description	Value	Unit
m_{UAV}	Mass of the UAV	4.2	kg
m_i	Link and Actuator mass	0.625	kg
I_{UAV}	Moment of Inertia (y) of the UAV	0.4097	kg·m^2
$I_{g,i}$	Moment of Inertia (y) of Link i	2.99×10^{-4}	kg·m^2
l_i	Link i length	0.13	m
a_i	Distance from joint i to link i CoG	0.117	m
x_0	Distance from UAV CoG to manipulator joint 1 along UAV body x-axis	0	m
z_0	Distance from UAV CoG to manipulator joint 1 along UAV body z-axis	-0.2	m

We use 3 different trajectories to be tracked by the manipulator end-effector in our simulated tests: 1. horizontal line, 2. tilted line, and 3. circle.

All trajectories have the same execution time (15 s). Trajectories 1–2 use a trapezoidal velocity profile, this velocity profile can be described as follows, for both the velocity components along x_I and z_I:

$$v = \begin{cases} v_{max} \cdot \left(\frac{t}{t_a}\right), & t < t_a \\ v_{max}, & t_a \leq t < T - t_a \\ v_{max} \cdot \left(1 - \frac{t-(T-t_a)}{t_a}\right), & T - t_a \leq t \end{cases} \tag{21}$$

where t_a is the acceleration (equal to the deceleration) time interval, T is the total time necessary to complete the whole trapezoid and v_{max} is the maximum velocity value (along x_I or z_I). The value of v_{max} (along x_I or z_I) can be calculated as:

$$v_{max} = \frac{\Delta s}{\|\boldsymbol{p}\|} \cdot a_{max} \cdot t_a \tag{22}$$

where Δs is the displacement along the axis (x or z), $\|\boldsymbol{p}\|$ is the magnitude of the total displacement and a_{max} is the maximum acceleration of the end-effector.

A detailed description of each trajectory is presented here below:

1. Horizontal line. Linear trajectory with a ± 0.1 m displacement along the x_I axis. The end-effector goes first to the right, then passes all the way through the left and then back again to the starting point. This trajectory is composed of a series of 3 stages, each one defined as a trapezoidal profile, as described in (21). All stages are executed in a sequential manner and have the same acceleration time $t_a = 0.79$ s and acceleration $a_{max} = 0.4$ m·s^{-2}. Table 2 shows the duration of each stage composing the trajectory, while Figure 4 shows the corresponding velocity profile and the initial configuration of the manipulator.

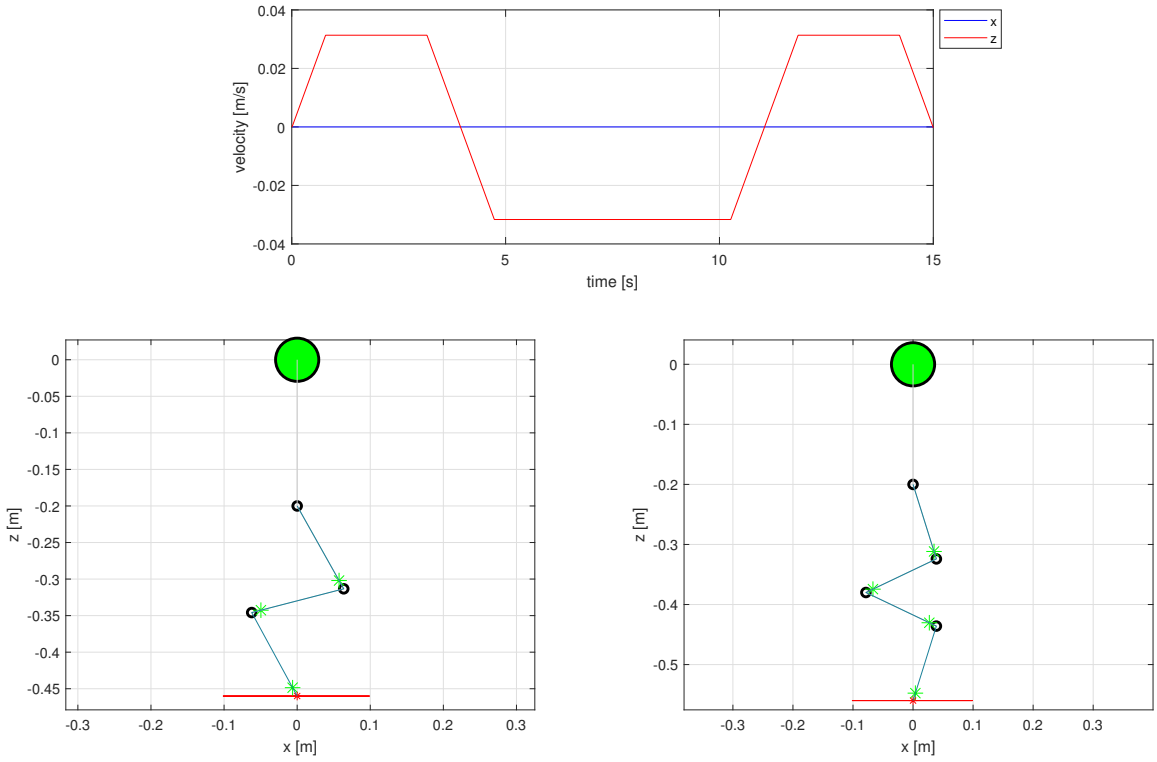

Figure 4. Trajectory 1 (Horizontal line) desired velocity profile for the end-effector (**top**). The initial configurations for the 3 DOF (**left**), and 4 DOF (**right**) manipulators are below, showing the UAV CoG (large green marker), joints (black circles), links (blue lines), link CoGs (green markers), end-effector initial position (red marker), and desired trajectory (red line).

Table 2. Duration of the three stages in the horizontal and tilted line trajectories.

Stage (i)	T [s]
1	3.95
2	7.10
3	3.95

2. Tilted line. Linear trajectory with a ± 0.1 m displacement along the x_I axis and ± 0.02 m along the z_I axis. The end-effector goes first to the right bottom side of the trajectory, goes back to the top left point, and finally returns to its starting position. Similarly to the previous trajectory, this one is composed of a series of 3 stages, each one defined as a trapezoidal profile, as described in (21). All stages are executed in a sequential manner and have the same acceleration time $t_a = 0.79$ s and acceleration $a_{max} = 0.4$ m·s^{-2}. Table 2 shows the time duration of each stage composing the trajectory, while Figure 5 shows the velocity profile of the trajectory and the initial configuration of the manipulator.

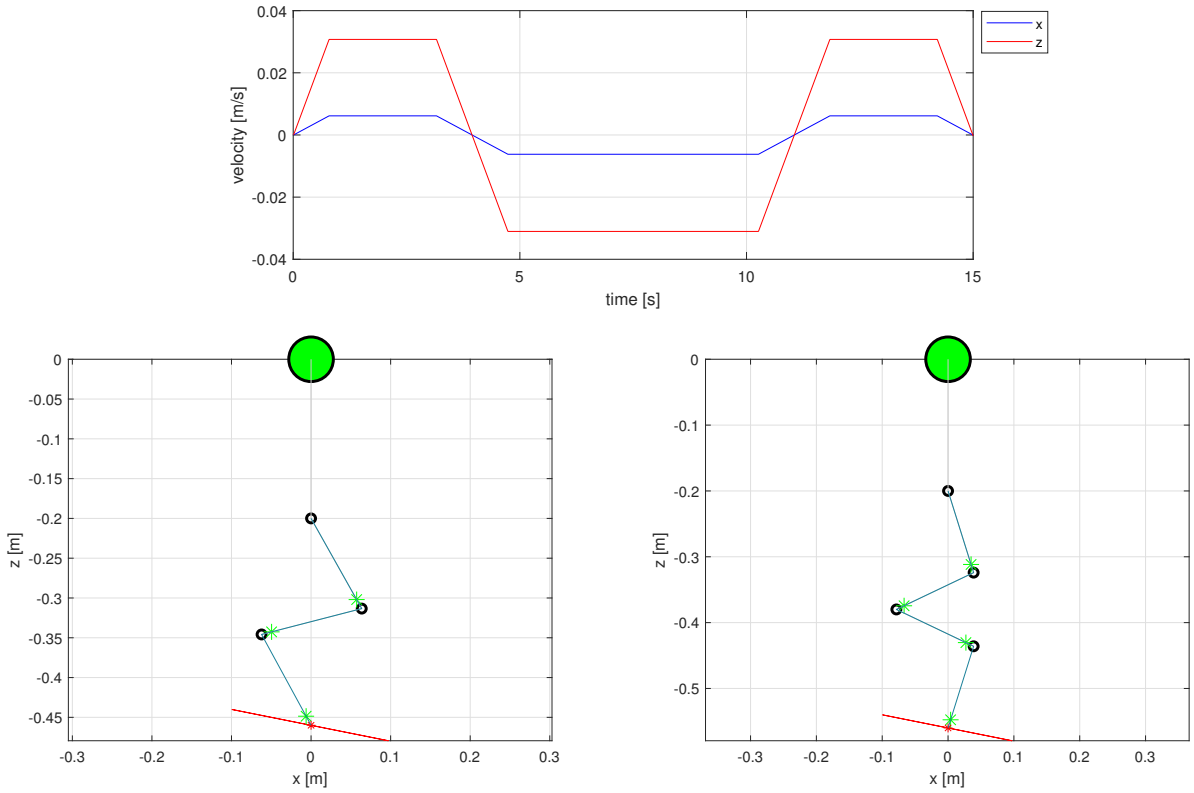

Figure 5. Trajectory 2 (Tilted line) desired velocity profile for the end-effector (**top**). The initial configurations for the 3 DOF (**left**), and 4 DOF (**right**) manipulators are below, showing the UAV CoG (large green marker), joints (black circles), links (blue lines), link CoGs (green markers), end-effector initial position (red marker), and desired trajectory (red line).

3. Circle. Circular trajectory with diameter 0.14 m, starting from the bottom of the circle. For more details about the generation of this profile please refer to [17]. Figure 6 shows the corresponding velocity profile for this trajectory, and the initial configuration of the manipulator.

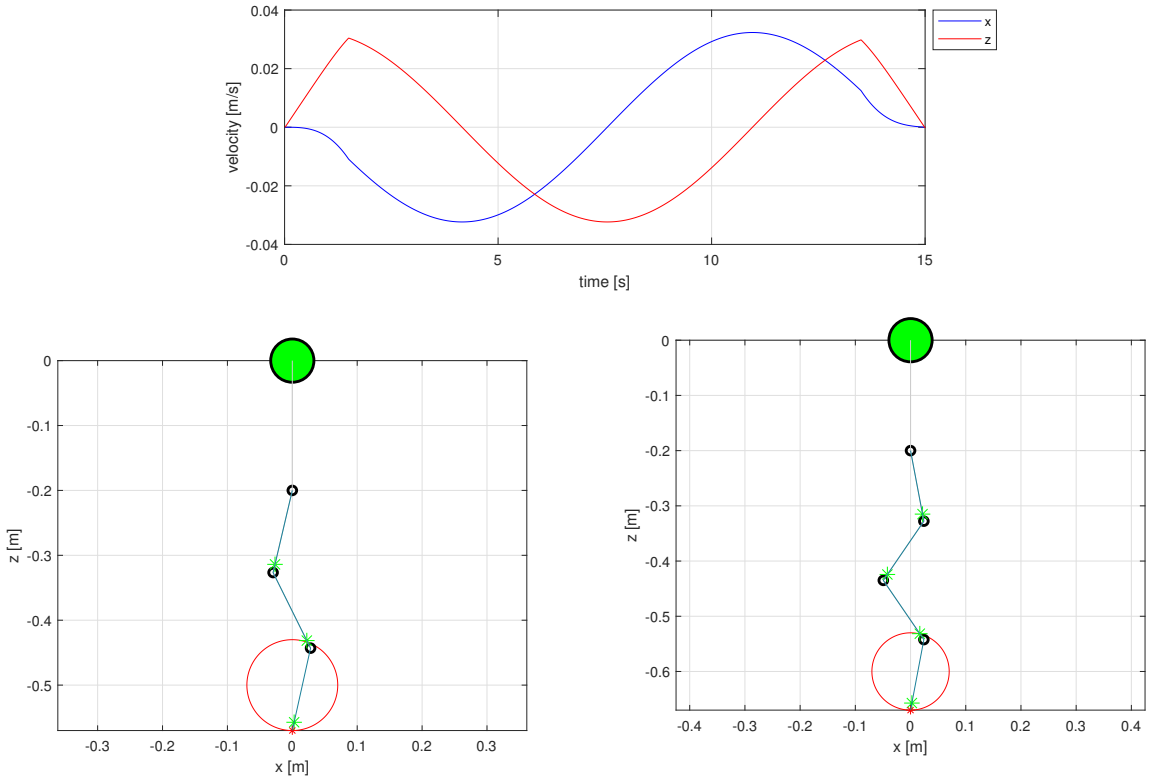

Figure 6. Trajectory 3 (Circle) desired velocity profile for the end-effector (**top**). The initial configurations for the 3 DOF (**left**), and 4 DOF (**right**) manipulators are below, showing the UAV CoG (large green marker), joints (black circles), links (blue lines), link CoGs (green markers), end-effector initial position (red marker), and desired trajectory (red line).

The PID controller coefficients utilized in the UAV control forces calculation (4)–(5) are summarized in Table 3.

Table 3. PID controller coefficients.

Coordinate (i)	k_{P_i}	k_{D_i}	k_{I_i}
z	37	18	8
ϕ	40	3	35

4.2. Results

Every trajectory is tracked with the 3 DOF and the 4 DOF aerial manipulator considering two cases for the inverse kinematics solution. We take the minimum joint velocities solution as a baseline for the inverse kinematics control of the manipulator (9). We then compare it with the gradient projection method that minimizes the horizontal translation of the manipulator CoG as outlined in Section 3.1.

1. Horizontal line.
   For the horizontal line trajectory tracked through the minimum joint velocities solution, the manipulator exerts a torque on the UAV at the beginning of the end-effector movement. This torque makes the UAV tilt to the right and, as a consequence, the UAV moves in that direction because of the horizontal component of the propeller's thrust. Along the whole trajectory, the UAV moves horizontally together with the end-effector with a small overshoot when the latter stops. Similar behavior also happens in the other two stages of the trajectory. Therefore, for both manipulators, joint angles show small variations (the difference between the maximum and minimum

angle of the same joint is smaller than 0.12 rad), and joint velocities are small (less than 0.1 rad·s^{-1}). The control torque of the UAV prevents excessive UAV rotation.
By tracking the same trajectory with the gradient projection method the UAV tilts and translates as for the minimum joint velocities solution. However, due to our secondary objective function in the inverse kinematic control, the maximum torque exerted by the manipulator on the UAV is reduced from 0.27 N·m to 0.13 N·m for the 3 DOF case, and from 0.27 N·m to 0.06 N·m for the 4 DOF case. As a result, the UAV rotation and translation decrease significantly.
The results for this trajectory tracking task are summarized in Table 4. It can be noticed that the gradient projection method gives a better improvement in the reduction of the horizontal displacement of the UAV during the first half of the trajectory task, thus reducing the maximum translation in the right direction more effectively. The 4 DOF UAM is of course better at reducing the horizontal displacement as compared to the 3 DOF, which is expected due to the kinematic redundancy coming from the additional degree of freedom.

Table 4. Maximum rotation and translation of the UAV using the minimum joint velocities solution (non-optimized) and the gradient projection method (optimized) for the horizontal line trajectory.

UAV Motion	3 DOF Non-Optimized	3 DOF Optimized	4 DOF Non-Optimized	4 DOF Optimized
Maximum rotation [rad]	6.1×10^{-3}	3.5×10^{-3}	9×10^{-3}	1.7×10^{-3}
Maximum translation on the right [m]	0.11	0.040	0.11	0.020
Maximum translation on the left [m]	0.11	0.049	0.11	0.035

The position error of the end-effector for both the manipulators is always less than 11 mm with the minimum joint velocities solution, and less than 3.6 mm using the gradient projection method solution. This confirms our assumption that we can neglect the effect of the base motion in the inverse kinematics step, in particular when we use the gradient projection method which minimizes the manipulator CoG motion. For the 3 DOF manipulator, Figures 7 and 8 show the stroboscopic views, the joint angles, the UAV translations and rotation, and the torque exerted by the manipulator on the UAV; results are shown both for the minimum joint velocities solution and for the gradient projection method. Figures 9 and 10 show the corresponding results for the 4 DOF manipulator.

2. Tilted line.
Similarly to the horizontal line trajectory, when solving the inverse kinematics with the minimum joint velocities solution the UAV tilts and translates in the horizontal direction following the end-effector movement. However, in this case, the joint angles show larger variations in order to achieve the vertical movement of the end-effector. The gradient projection method reduces the torque disturbance compared to the minimum joint velocities solution, and as a result, the UAV rotation and horizontal translation are also reduced significantly. The torque disturbance is reduced from 0.26 N·m to 0.012 N·m for the 3 DOF manipulator, and from 0.28 N·m to 0.069 N·m for the 4 DOF case. These results are summarized in Table 5.

Table 5. Maximum rotation and translation of the UAV using the minimum joint velocities solution (non-optimized) and the gradient projection method (optimized) for the tilted line trajectory.

UAV Motion	3 DOF Non-Optimized	3 DOF Optimized	4 DOF Non-Optimized	4 DOF Optimized
Maximum rotation [rad]	6×10^{-3}	3.2×10^{-3}	9.6×10^{-3}	1.8×10^{-3}
Maximum translation on the right [m]	0.11	0.039	0.11	0.021
Maximum translation on the left [m]	0.11	0.048	0.11	0.034

Similarly to the horizontal line trajectory, it can be noticed that the gradient projection method gives a better improvement in the first part of the trajectory.
The position error of the end-effector is also smaller when using the gradient projection method: less than 3.4 mm in comparison to 11.9 mm for the minimum joint velocities solution.
For the 3 DOF manipulator, Figures 11 and 12 show the stroboscopic views, the joint angles, the UAV translations and rotation, and the torque disturbance exerted by the manipulator on the UAV; results are shown both for the minimum joint velocities solution and for the gradient projection method. Figures 13 and 14 show the same results for the 4 DOF manipulator.

3. Circle.
By tracking the circle trajectory with the gradient projection method, the maximum torque applied by the manipulator on the UAV decreases from 0.2 (minimum joint velocities solution) to 0.05 N·m for the 3 DOF UAM, and from 0.19 N·m to 0.05 N·m for the 4 DOF UAM. For both manipulators, the gradient projection method is also more effective in reducing the maximum roll angle for this trajectory. However, for 3 DOF UAM, the UAV translation is reduced effectively only in the first part of the motion (right direction), while the maximum UAV displacement on the left side is almost unchanged. Furthermore, for the 4 DOF manipulator the improvement in UAV translation is more noticeable for the first part of the trajectory. However, also the maximum displacement in the left direction shows a 50% reduction compared to the minimum joint velocities solution. Maximum rotation and translations of the UAV are reported in Table 6.

Table 6. Maximum rotation and translation of the UAV using the minimum joint velocities solution (non-optimized) and the gradient projection method (optimized) for the circular trajectory.

UAV Motion	3 DOF Non-Optimized	3 DOF Optimized	4 DOF Non-Optimized	4 DOF Optimized
Maximum rotation [rad]	3.5×10^{-3}	1.1×10^{-3}	4.2×10^{-3}	0.73×10^{-3}
Maximum translation on the right [m]	0.091	0.0093	0.041	0.0030
Maximum translation on the left [m]	0.059	0.058	0.13	0.060

The position error of the end-effector is always smaller than 6.2 mm when using the minimum joint velocities solution, and smaller than 1.7 mm using the gradient projection method.

For the 3 DOF manipulator, the stroboscopic views, the joint angles, the UAV translations and rotation, and the torque disturbance exerted by the manipulator on the UAV are shown in Figures 15 and 16. The results are shown both for the minimum joint velocities solution and for the gradient projection method. The same results are shown in Figures 17 and 18 for the 4 DOF manipulator.

As observed in the stroboscopic figures for the simulations, there is a noticeable difference in the end-effector position error between the non-optimized (minimum joint velocities) and the optimized (gradient projection method) case. This is because our inverse kinematic algorithm does not account for the base motions caused during the kinematic inversion, but it corrects them in the next step.

The optimized case reduces significantly this error as base displacements are minimized. The results for the maximum end-effector position error for all the simulation experiments are summarized in Table 7.

Computational time was measured during the simulations. The hardware used was equipped with an Intel i7 eighth generation exa-core processor, 16 GB RAM, and SSD mass storage. The operating system was Windows 11 and the MATLAB version was 2022a. The optimized case simulations took an average of 9.60 s with ± 0.27 s standard deviation for the 3 DOF arm and an average of 11.98 s with ± 0.15 s standard deviation for the 4 DOF

configuration. This is less than the trajectory time of 15 s, thereby enabling the real-time implementation of our algorithm in a physical aerial manipulator system.

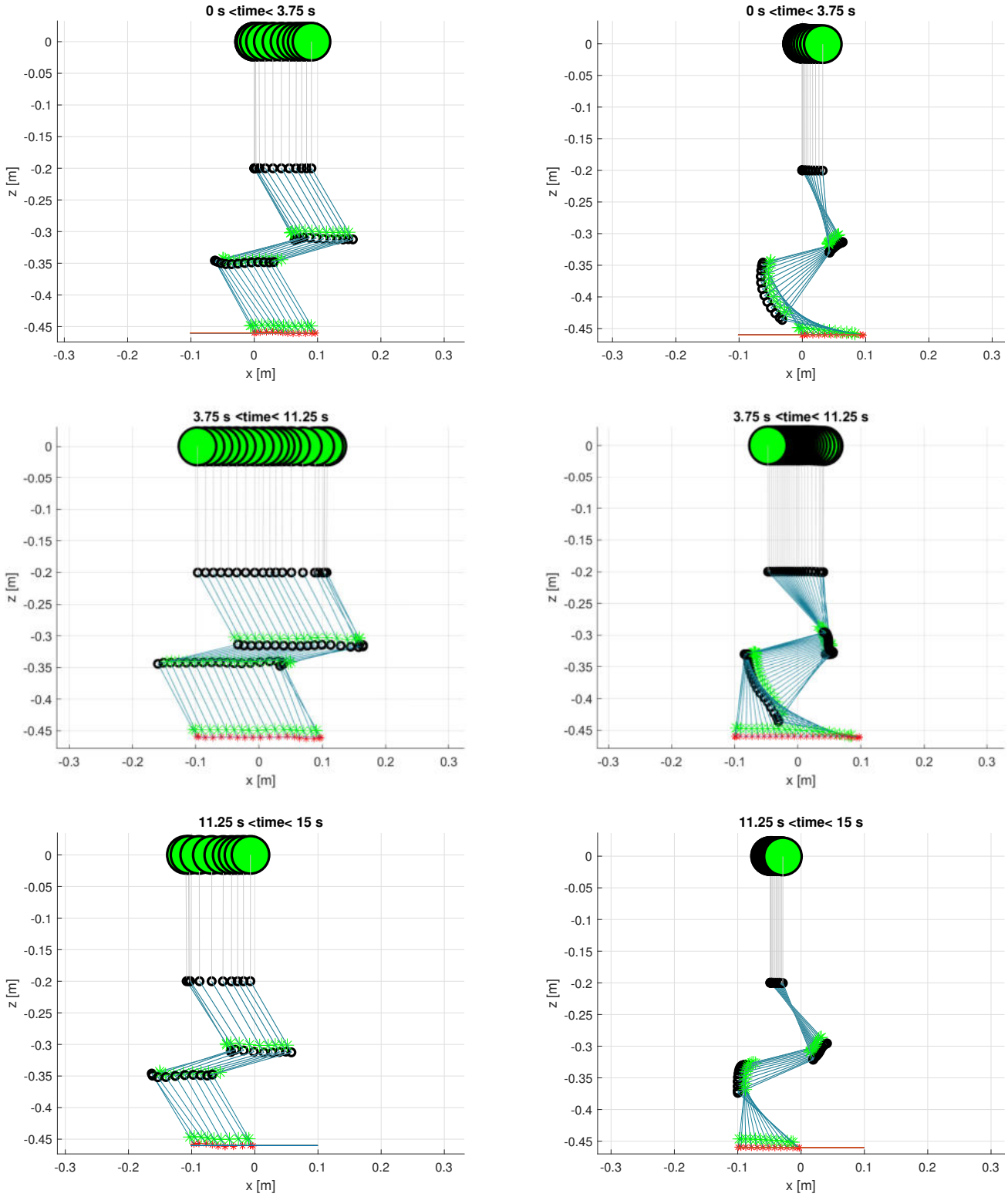

Figure 7. Stroboscopic view of the 3 stages of the horizontal line trajectory for the 3 DOF manipulator with the minimum joint velocities solution (**left**) and with the gradient projection method (**right**). The UAV CoG is the large green marker, joints are the black circles, links are the blue lines, link CoGs are green markers, and end-effector positions tracked are red markers.

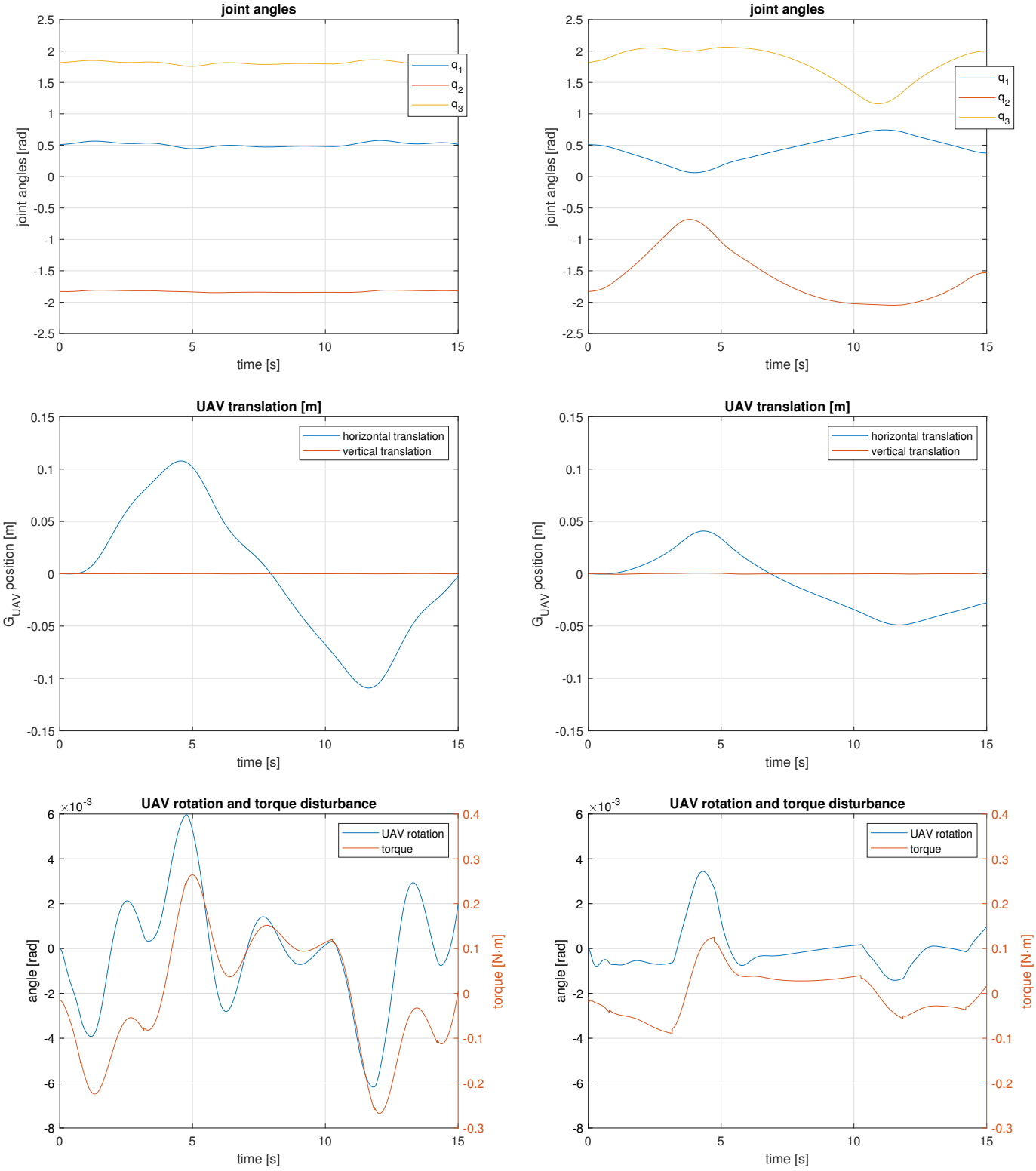

Figure 8. Manipulator joint motions (**top**) and UAV motions and torque disturbance (**middle** and **bottom**) for the 3 DOF UAM tracking the horizontal line trajectory. Results obtained with the minimum joint velocities solution are on the **left** and results obtained with the gradient projection method are on the **right**.

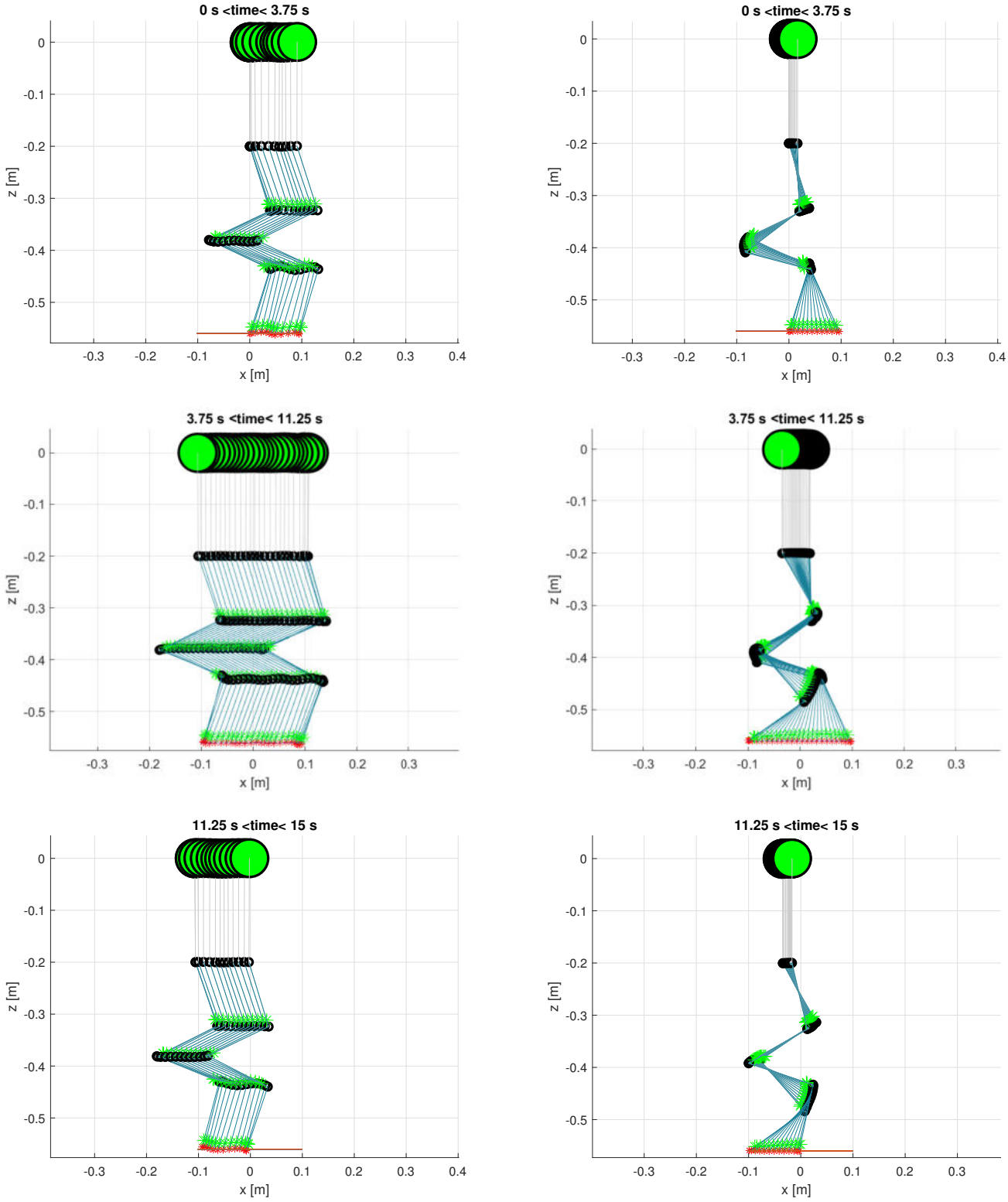

Figure 9. Stroboscopic view of the 3 stages of the horizontal line trajectory for the 4 DOF manipulator with the minimum joint velocities solution (**left**) and with the gradient projection method (**right**). The UAV CoG is the large green marker, joints are the black circles, links are the blue lines, link CoGs are green markers, and end-effector positions tracked are red markers.

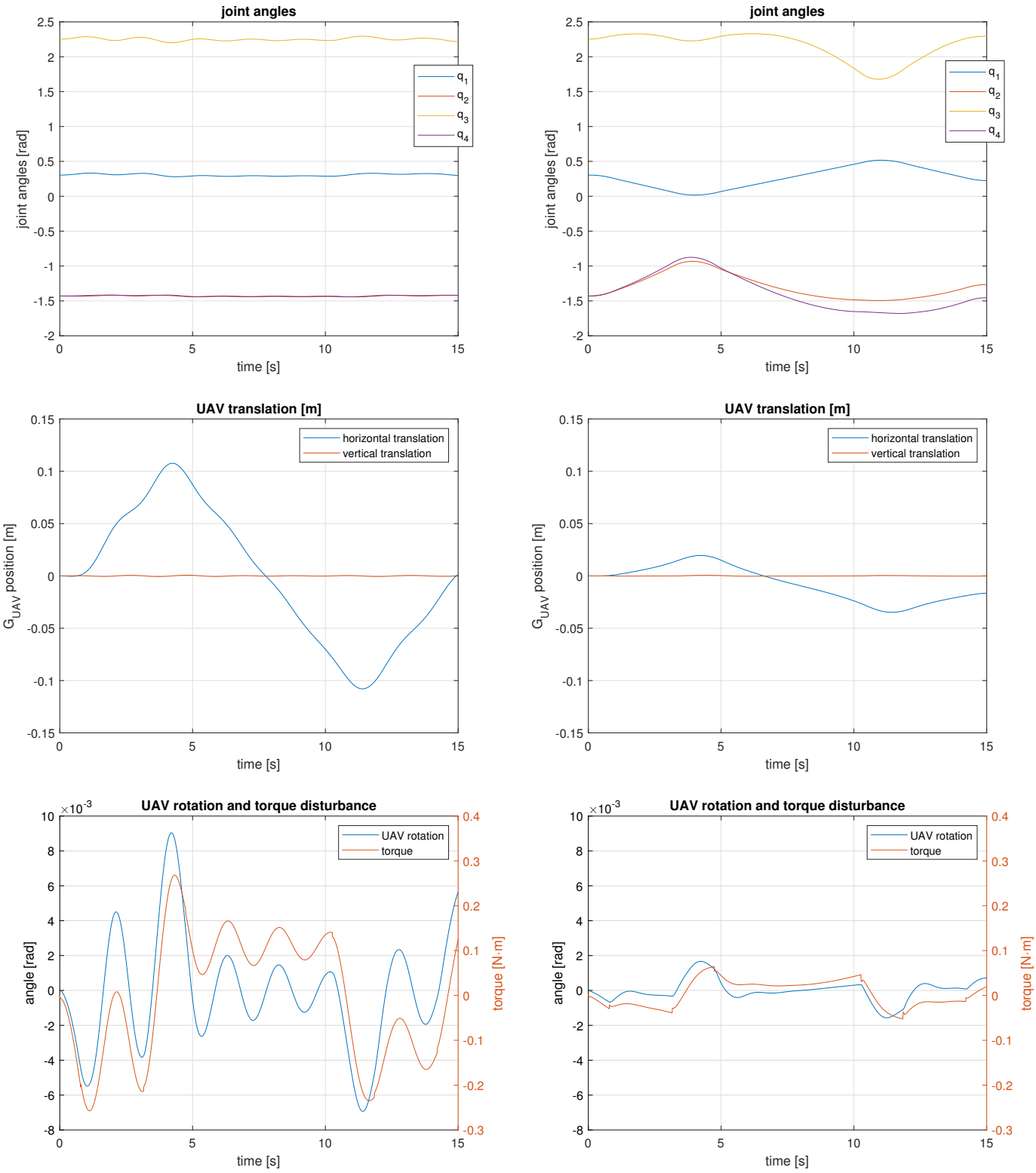

Figure 10. Manipulator joint motions (**top**) and UAV motions and torque disturbance (**middle** and **bottom**) for the 4 DOF UAM tracking the horizontal line trajectory. Results obtained with the minimum joint velocities solution are on the **left** and results obtained with the gradient projection method are on the **right**.

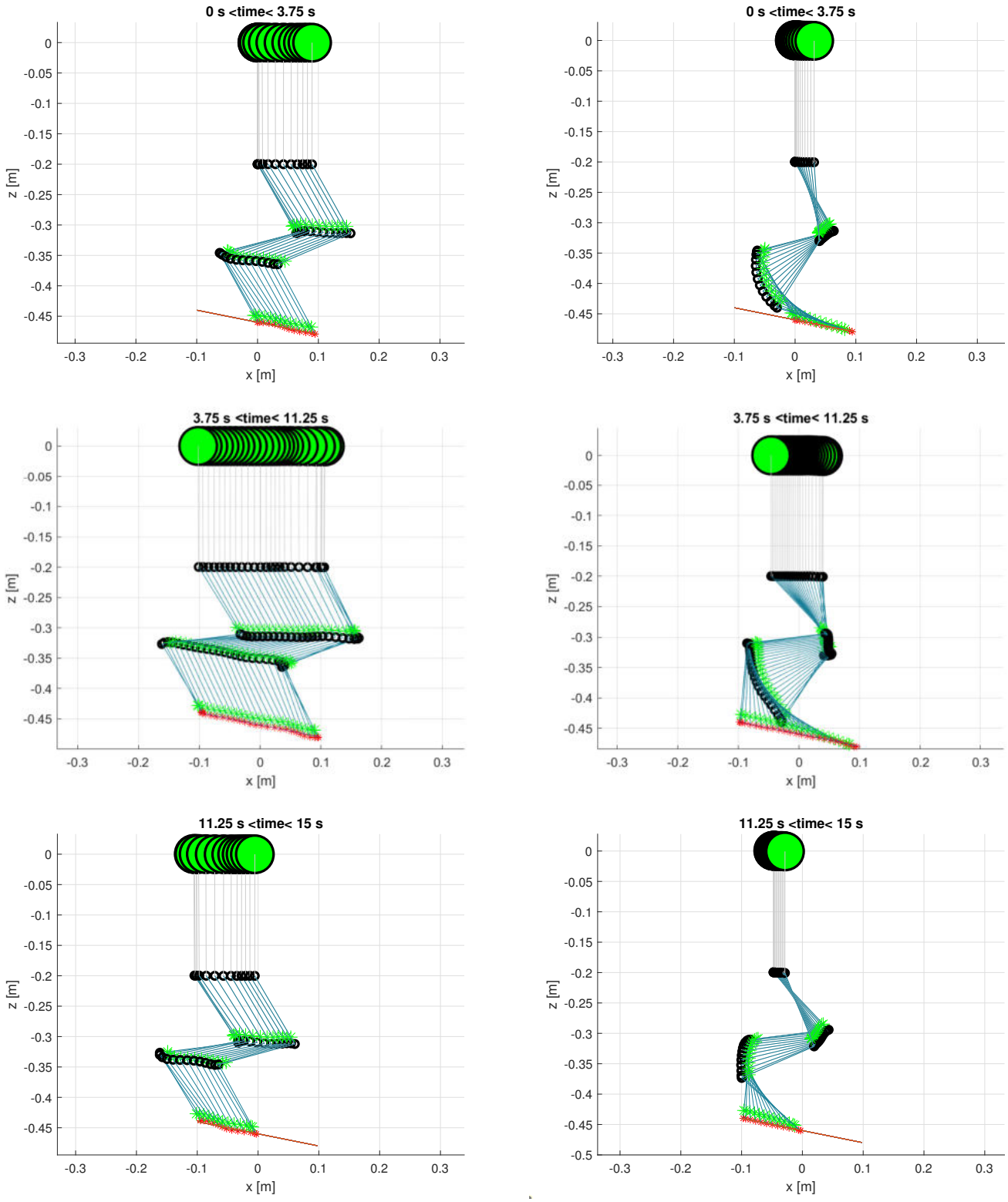

Figure 11. Stroboscopic view of the 3 stages of the tilted line trajectory for the 3 DOF manipulator with the minimum joint velocities solution (**left**) and with the gradient projection method (**right**). The UAV CoG is the large green marker, joints are the black circles, links are the blue lines, link CoGs are green markers, and end-effector positions tracked are red markers.

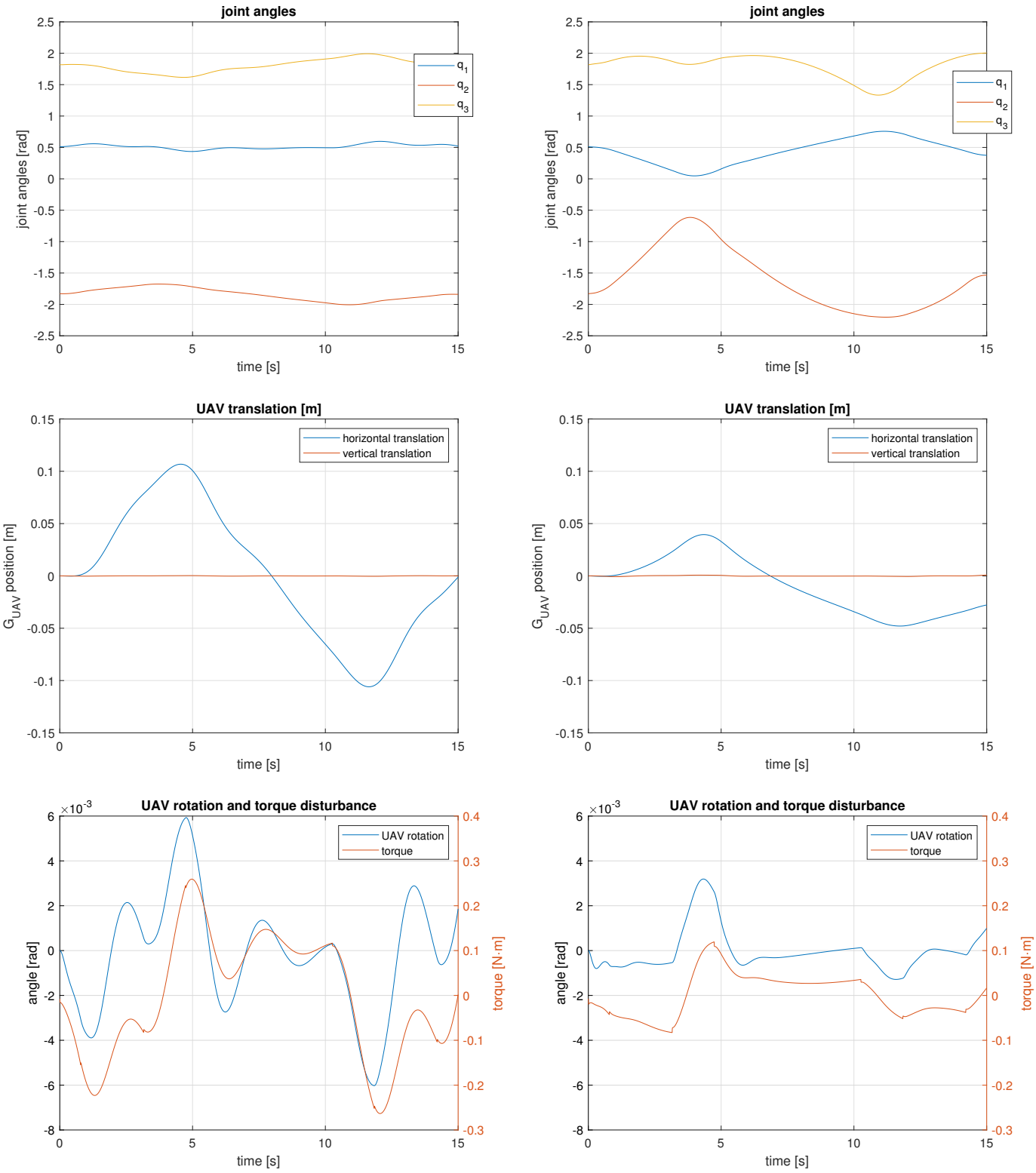

Figure 12. Manipulator joint motions (**top**) and UAV motions and torque disturbance (**middle** and **bottom**) for the 3 DOF UAM tracking the tilted line trajectory. Results obtained with the minimum joint velocities solution are on the **left** and results obtained with the gradient projection method are on the **right**.

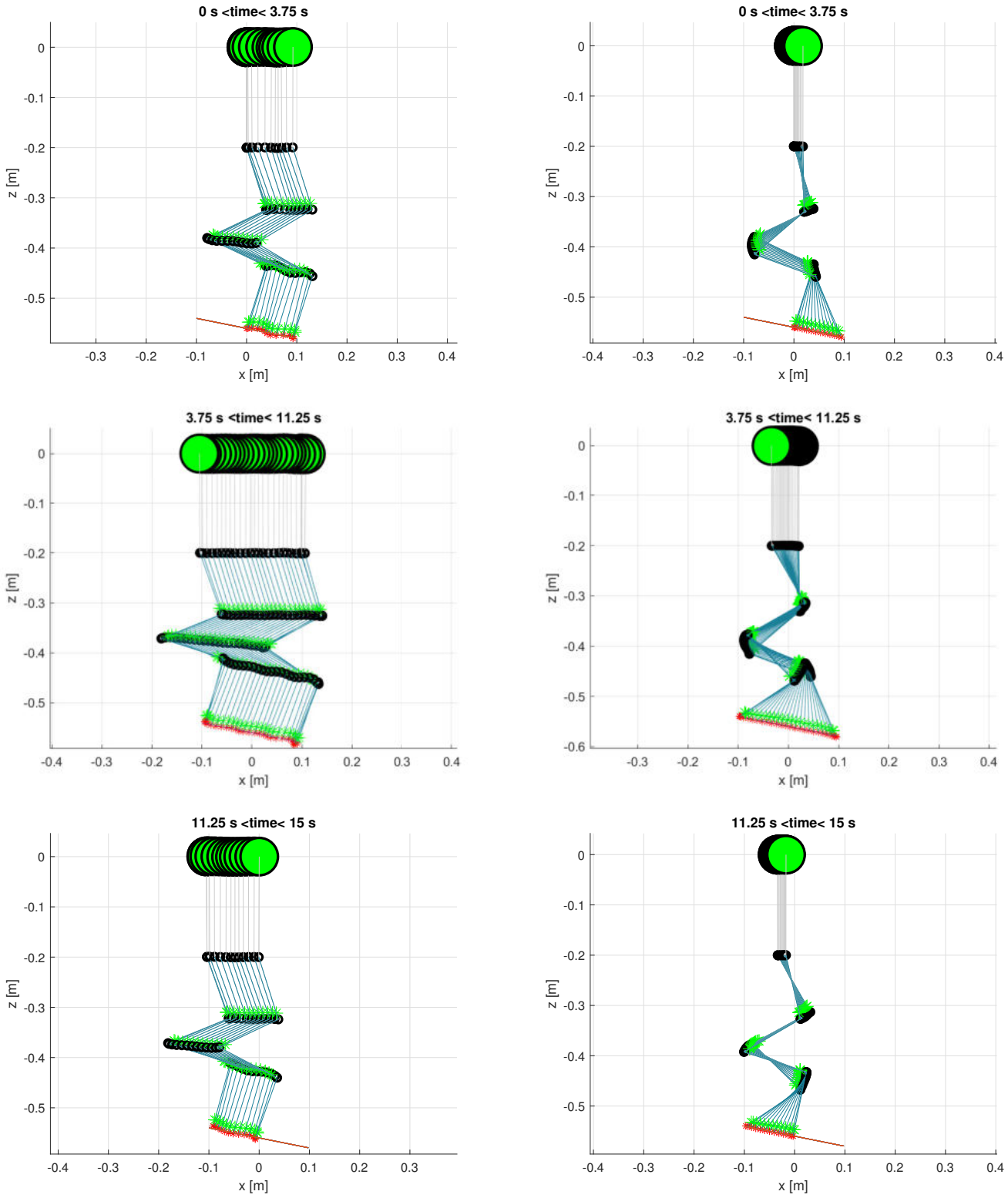

Figure 13. Stroboscopic view of the 3 stages of the tilted line trajectory for the 4 DOF manipulator with the minimum joint velocities solution (**left**) and with the gradient projection method (**right**). The UAV CoG is the large green marker, joints are the black circles, links are the blue lines, link CoGs are green markers, and end-effector positions tracked are red markers.

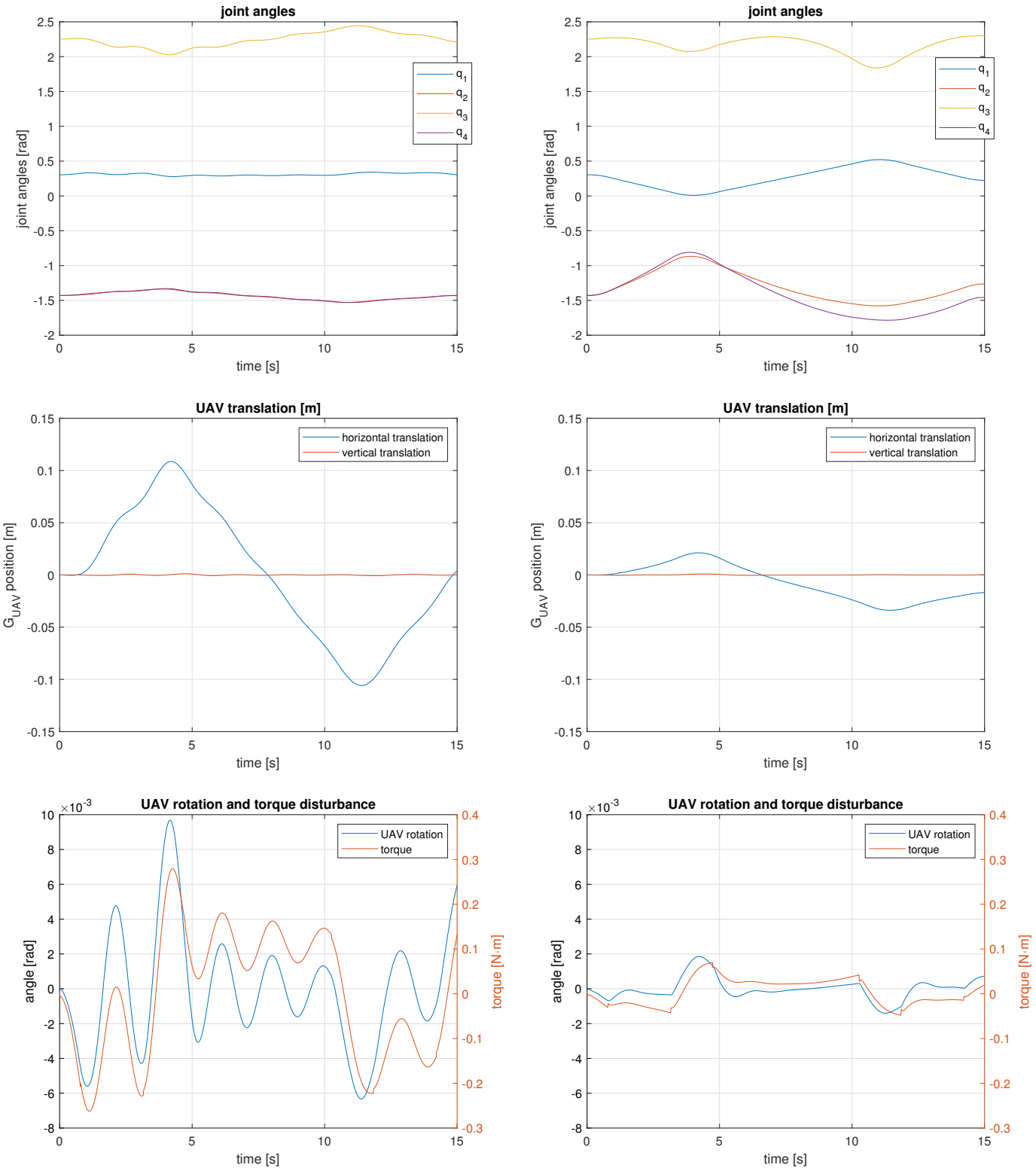

Figure 14. Manipulator joint motions (**top**) and UAV motions and torque disturbance (**middle** and **bottom**) for the 4 DOF UAM tracking the tilted line trajectory. Results obtained with the minimum joint velocities solution are on the **left** and results obtained with the gradient projection method are on the **right**.

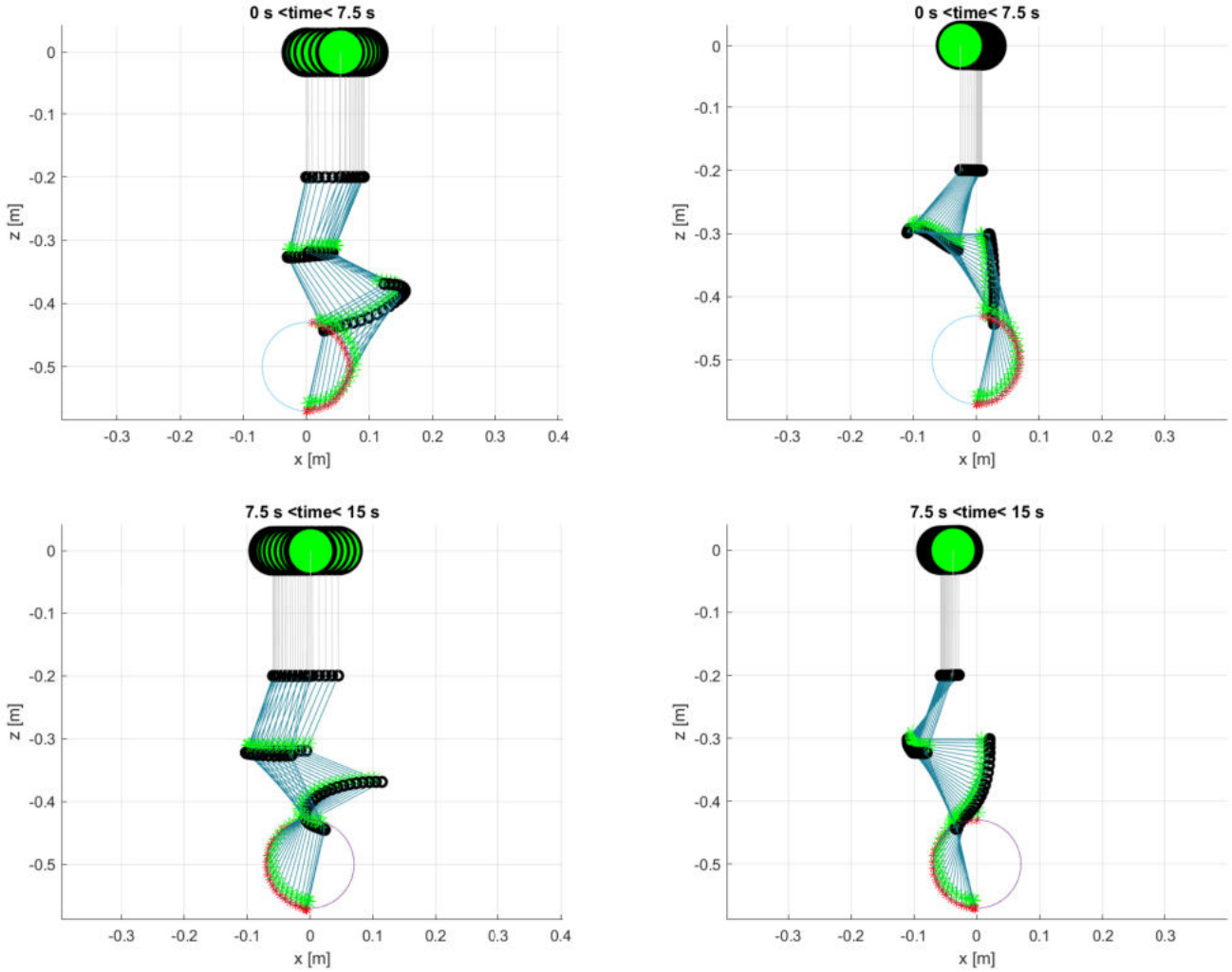

Figure 15. Stroboscopic view of the 2 stages of the circle trajectory for the 3 DOF manipulator with the minimum joint velocities solution (**left**) and with the gradient projection method (**right**). The UAV CoG is the large green marker, joints are the black circles, links are the blue lines, link CoGs are green markers, and end-effector positions tracked are red markers.

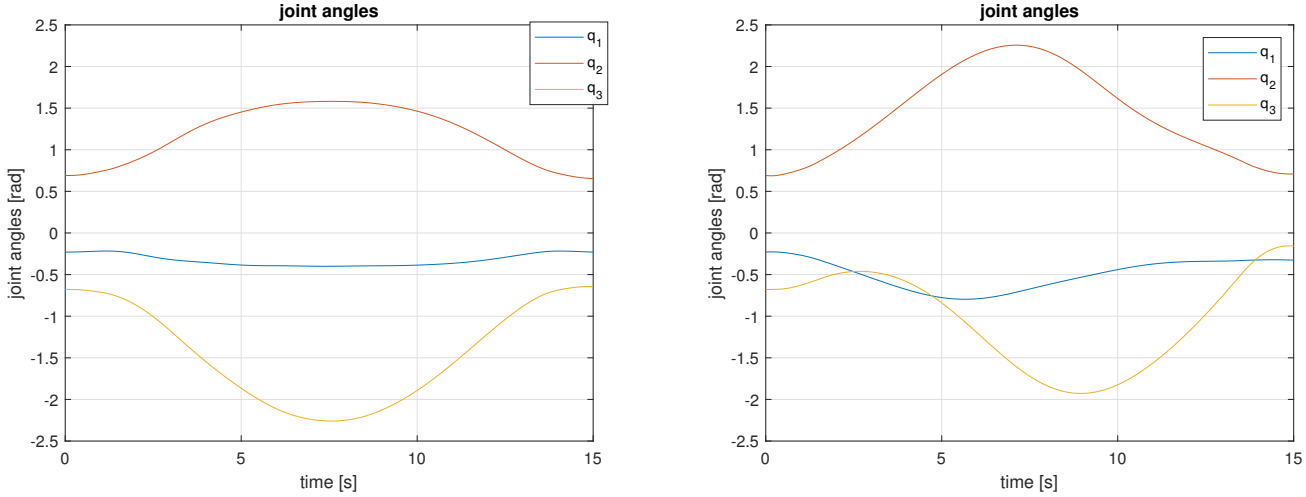

Figure 16. *Cont.*

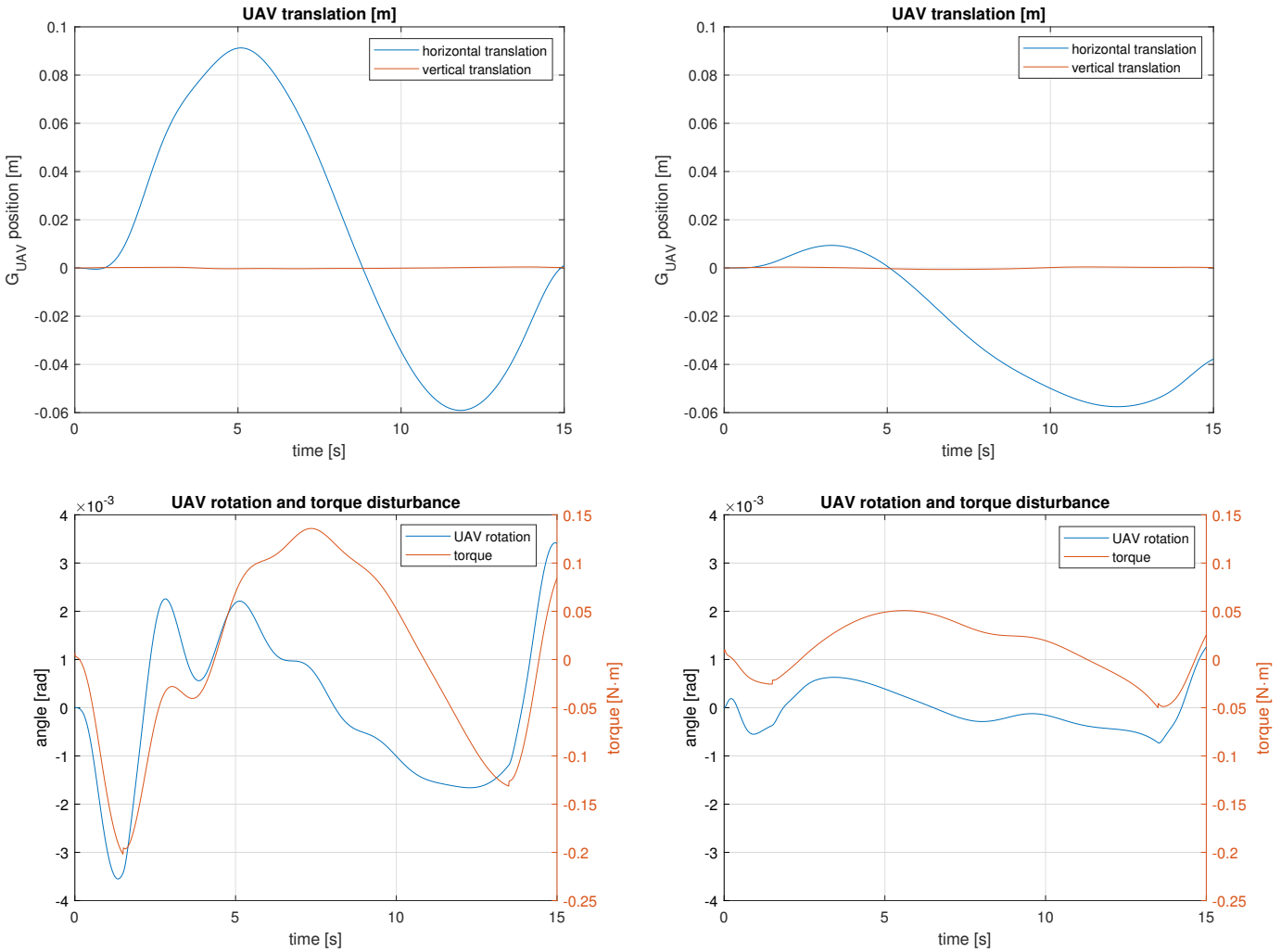

Figure 16. Manipulator joint motions (**top**) and UAV motions and torque disturbance (**middle** and **bottom**) for the 3 DOF UAM tracking the circle trajectory. Results obtained with the minimum joint velocities solution are on the **left** and results obtained with the gradient projection method are on the **right**.

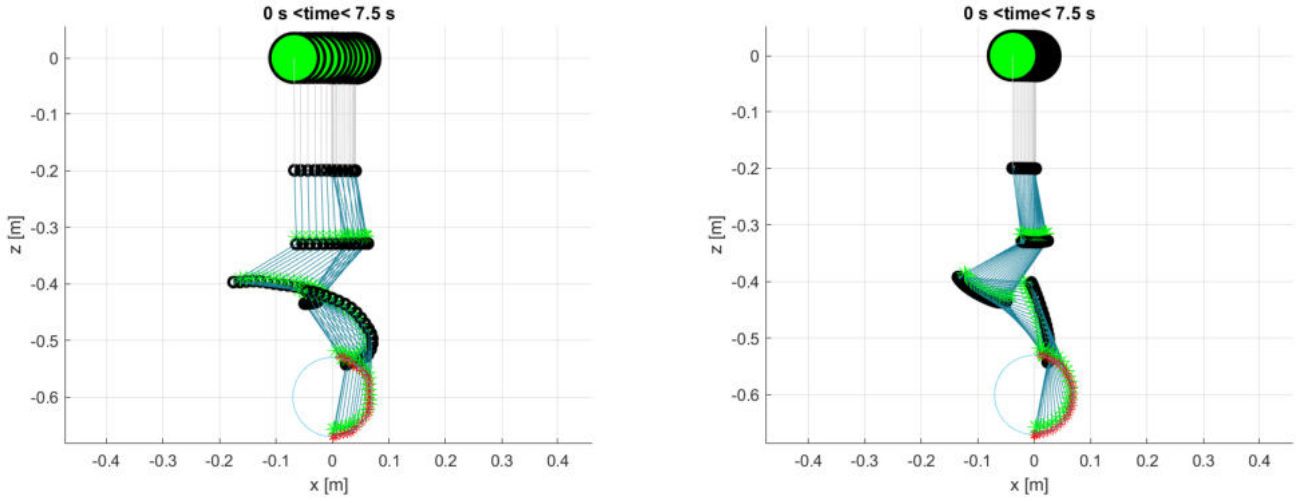

Figure 17. *Cont.*

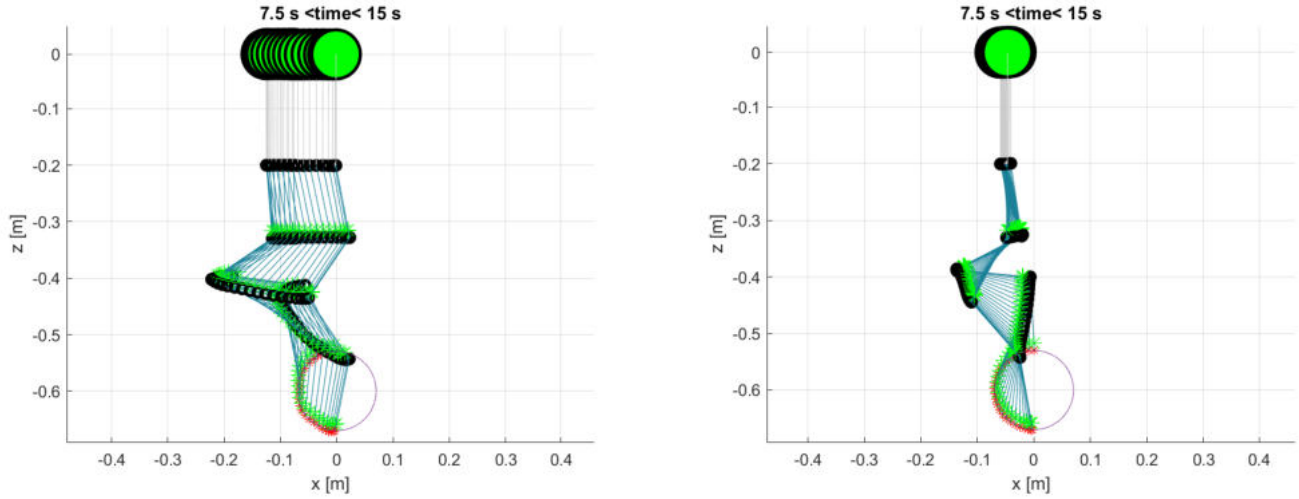

Figure 17. Stroboscopic view of the 2 stages of the circle trajectory for the 4 DOF manipulator with the minimum joint velocities solution (**left**) and with the gradient projection method (**right**). The UAV CoG is the large green marker, joints are the black circles, links are the blue lines, link CoGs are green markers, and end-effector positions tracked are red markers.

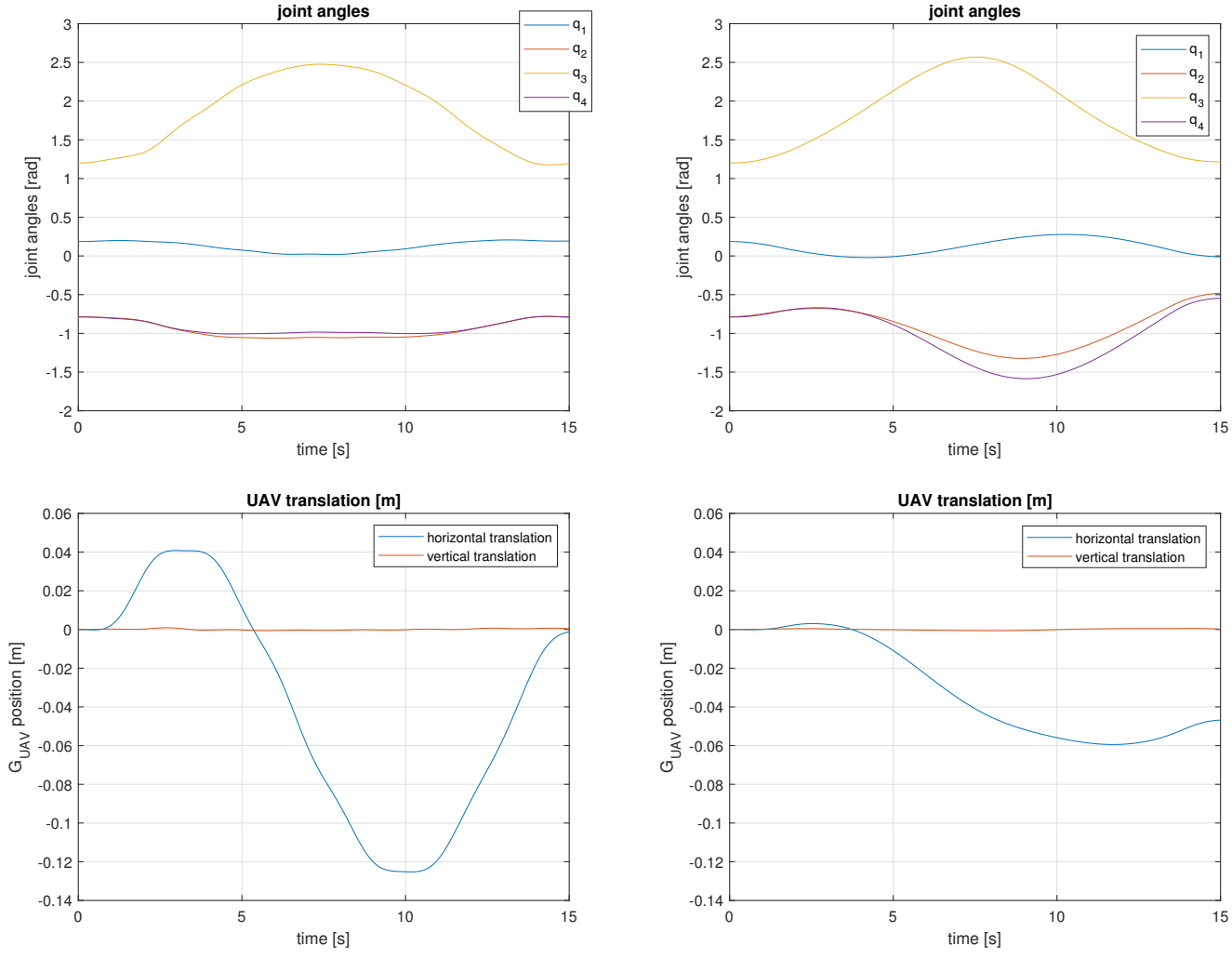

Figure 18. *Cont.*

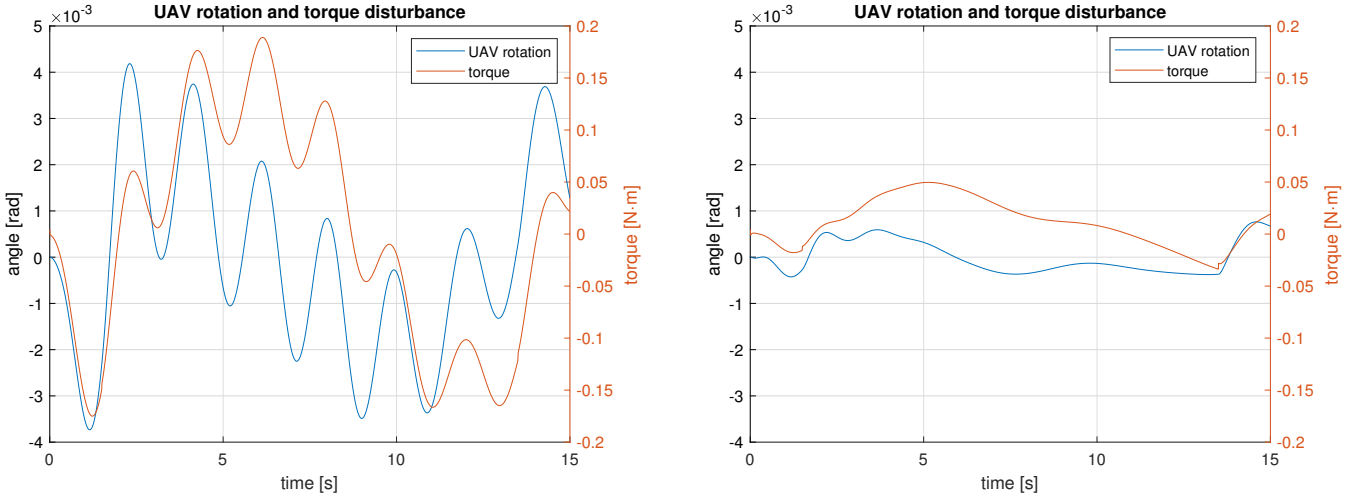

Figure 18. Manipulator joint motions (**top**) and UAV motions and torque disturbance (**middle** and **bottom**) for the 4 DOF UAM tracking the circle trajectory. Results obtained with the minimum joint velocities solution are on the **left** and results obtained with the gradient projection method are on the **right**.

Table 7. End-effector maximum position error for all tested trajectories and arm configurations.

Trajectory (*i*)	Maximum End-Effector Position Error [mm]			
	3 DOF Non-Optimized	3 DOF Optimized	4 DOF Non-Optimized	4 DOF Optimized
1 (horizontal line)	6.6	3.6	11.0	2.1
2 (tilted line)	6.3	3.4	11.9	2.4
3 (circle)	4.7	1.7	6.2	1.2

5. Workspace Computation

5.1. Simulation Setup for Workspace Computation

For a manipulator attached to a UAV in a hovering flight, the workspace is not limited only by joint limits, robot architecture and initial configuration. The UAV motion due to the joint motion and resultant controller response can also increase the distance between the UAV and the desired position of the end-effector. The gradient projection method can be used to reduce the horizontal translation of the UAV as shown in Sections 3 and 4. However, when the trajectory tracked by the end-effector is excessively long, the gradient projection method is no longer able to keep the UAV translations small. Hence we augment the conventional workspace analysis with a bound for the maximum acceptable UAV translation.

In order to scan the workspace we define a straight trajectory for the end-effector. The trajectory is divided into a first part with constant acceleration and a part with a constant velocity of indefinite duration, similar to (21) but without the deceleration phase. Starting from the initial equilibrium configuration of the UAM, this trajectory is executed in all directions by rotating it around the end-effector initial point: 720 equally-spaced lines are used.

The algorithm described in Section 3.2 is applied to simulate each of these trajectories. Since the desired trajectories have infinite lengths the simulation is stopped when joint velocities exceed a threshold value (i.e., the manipulator is approaching a singular configuration). In this analysis, joint limits are neglected. A similar procedure was also used in [17].

Moreover, we are interested in the workspace that can be reached without large UAV motions, so the simulation is also stopped when the horizontal translation of the UAV exceeds a threshold value.

The simulation results in Section 4.2 showed that the position error of the end-effector can be large, in particular for the minimum joint velocities solution. Therefore, we included a stopping condition for excessive position error of the end-effector. In this case, we choose a threshold value that is between the maximum error measured with the gradient projection method and with the minimum joint velocities solution.

The threshold values used are 2 rad·s^{-1} for joint velocities, 0.02 m for the horizontal translation of the UAV, and 5 mm for the end-effector position error. The study was carried out both for the 3 DOF and the 4 DOF manipulator.

5.2. Workspace Simulation Results

The workspace limits computed are reported in Figure 19 with the stopping criteria type for each trajectory indicated with different colored markers.

For both the 3 DOF and the 4 DOF manipulator, we notice that the workspace is not symmetric, as our initial conditions result in path dependency for the trajectories determining the workspace limits. Comparing the positive and negative x-axis limit, there is a difference both in the distance of the workspace limit from the z-axis and in the stopping condition of the simulation. In particular, the UAV translation threshold defines the negative x-axis limit of the workspace. The positive x-axis limits are mostly determined by the UAV translation stopping condition and by the joint velocity limit stopping condition. An inverted initial configuration about $x = 0$ would also invert the workspace limits.

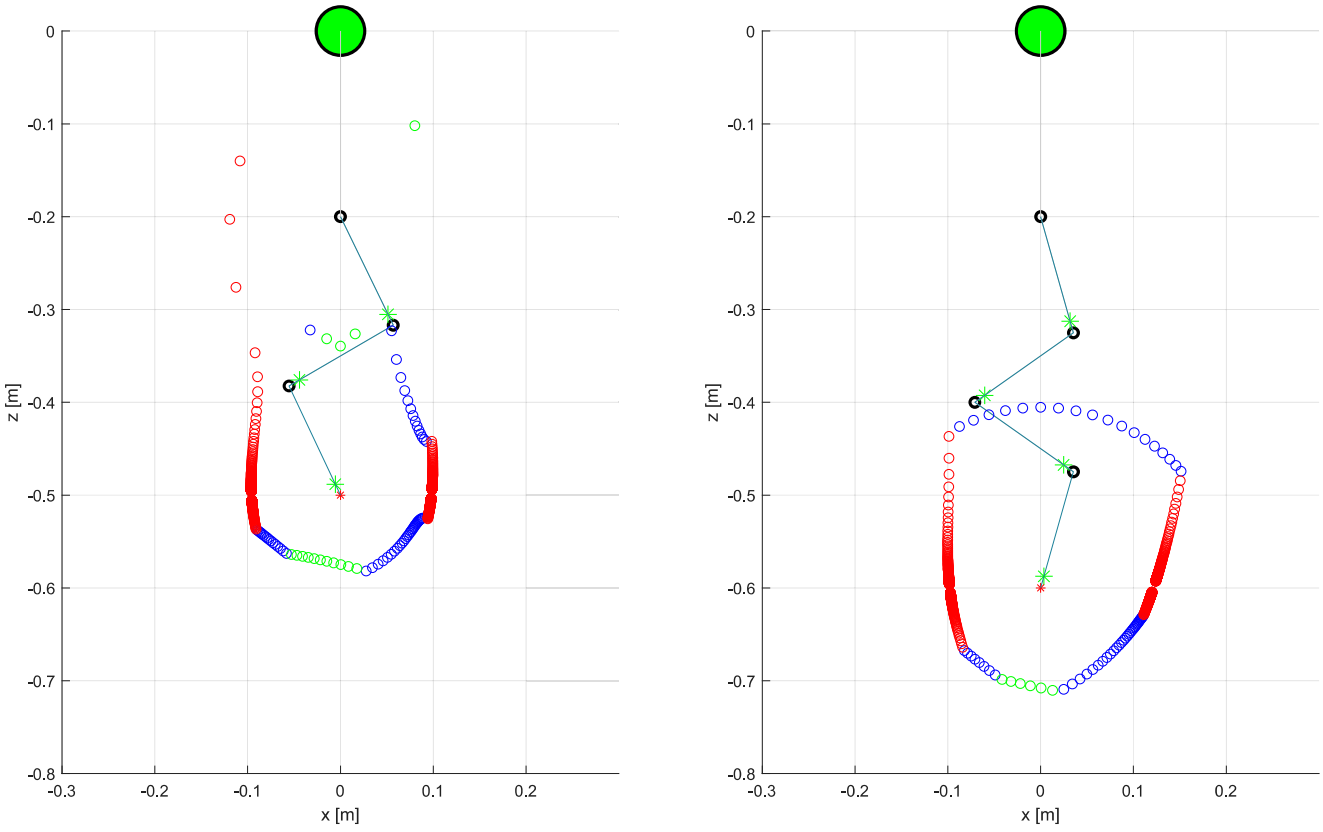

Figure 19. Workspace limits for the 3 DOF (**left**) and for the 4 DOF (**right**) UAM. Markers indicate the stopping position while color refers to the stopping condition: green for excessive joint velocities, red for excessive UAV translation, and blue for excessive position error. The UAM initial configuration is shown through the UAV CoG (large green marker), joints (black circles), links (blue lines), link CoG (green marker), and end-effector position (red marker).

For both the 3 and 4 DOF manipulators, the bottom limit is partially defined by the threshold velocity of joints (i.e., the simulation stopped because of a singular configuration) and by the end-effector position error limit. By looking at the 4 DOF manipulator, the workspace is more extended for trajectories in the upper direction due to the additional link length and DOF. Contrarily, in the case of the 3 DOF UAM the upper workspace boundary is not as well defined. Nevertheless, standard joint angle limits and manipulator mount design also impose natural restrictions on this kind of configuration.

As expected the workspace is wider for the 4 DOF manipulator. However, from the 3 DOF to the 4 DOF manipulator, the workspace is increased only in some directions. In particular, the most noticeable improvement is for trajectories in the positive x-axis direction. At the workspace limit along the z-axis the links are almost aligned, so the additional link in the 4 DOF manipulator further extends the workspace in this direction. For other directions, the workspace improvement is not particularly significant as the horizontal translation of the UAV causes excessive end-effector position error.

6. Conclusions

The main contribution of this work is the use of null-space projection to minimize a kinematically redundant manipulator CoG displacement while tracking a trajectory with an end-effector. As expected, this resulted in reduced base displacement in comparison to the non-optimized inverse kinematic control algorithm that minimizes joint velocities only.

Our simulation results, presented in Section 4.2 show that on average we are able to reduce the maximum horizontal base displacement by 56% across all trajectories, for both the 3 and 4 DOF manipulator, with a better improvement for the 4 DOF manipulator. In addition to that, we see improved precision, measured through a reduction in the end-effector position error between the non-optimized and optimized control algorithm simulations of about 66% averaged across all trajectories. This is an expected result as minimizing UAV base displacements reduces the error in the motion of the end-effector during the kinematic inversion stage (which is corrected only in the following timestep).

The algorithm runs on average in 11 s for a trajectory time of 15 s, on our available hardware. The computation time is less than the trajectory time, enabling real-time application. It is important to note that our code is not optimized, and also includes the complexity of calculating the forward dynamics to estimate the base motion. In a real-world implementation, the required base (and joints) state can be obtained through sensors, and the only step running would be the inverse kinematic control in an optimized code. Therefore it is likely that an onboard microcontroller with the algorithm implemented in C or C++ will provide adequate computing power for a real-world manipulator completing this type of task.

We also conducted a workspace computation on both the 3 and 4 DOF manipulators, which also included a stopping criterion for excessive base motions. As expected, the 4 DOF has a greater workspace compared to the 3 DOF. However, the main restriction on the workspace comes from base displacements being exceeded. Our inverse kinematic control algorithm neglects the direct effects of the base motion in the joint control, which could be taken into account using a generalized Jacobian [10,12,23].

Future work to continue this research includes testing on a developed real-world platform, which requires access to accurate real-time positioning of the UAM, through the means of a custom visual localization system, such as a Vicon motion system.

We plan on building an initial prototype by mounting a 6 DOF redundant arm on our DJI S1000 UAV, and conducting these experiments in a live setting. The arm will have servomotors that allow us to control the joint velocities at required speeds (up to 1–2 $\text{rad}\cdot\text{s}^{-1}$) and have a workspace of about 0.4–0.5 m along the x axis to track similar trajectories to those in the simulations. The arm should meet certain other specifications in terms of weight (less than 3 kg) and size (to mount below the main frame). We will start with experiments in the 2D plane, and then we will expand this to 3D trajectories. Finally, we

plan to build custom arms with greater redundancy to further test the proposed dynamic balancing algorithm.

Author Contributions: Conceptualization, S.C.; methodology, S.C. and N.M.; software, A.P. and N.M.; investigation, S.C., A.P., Y.V., V.A.-A. and N.M.; writing—original draft preparation, Y.V., A.P. and V.A.-A.; writing—review and editing, S.C., Y.V., A.P. and V.A.-A.; supervision, project administration, funding acquisition, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was carried out in the framework of the project "DynAeRobot—Development and validation of a new dynamically balanced aerial manipulator" (project no. BIRD213590), funded under the BIRD 2021 program promoted by the University of Padova.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CoG	Center of Gravity
DOF	Degree of Freedom
UAM	Unmanned Aerial Manipulator
UAV	Unmanned Aerial Vehicle

References

1. Ollero, A.; Tognon, M.; Suarez, A.; Lee, D.; Franchi, A. Past, present, and future of aerial robotic manipulators. *IEEE Trans. Robot.* **2021**, *38*, 626–645. [CrossRef]
2. Ohnishi, Y.; Takaki, T.; Aoyama, T.; Ishii, I. Development of a 4-joint 3-DOF robotic arm with anti-reaction force mechanism for a multicopter. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 985–991.
3. Trujillo, M.Á.; Martínez-de Dios, J.R.; Martín, C.; Viguria, A.; Ollero, A. Novel aerial manipulator for accurate and robust industrial NDT contact inspection: A new tool for the oil and gas inspection industry. *Sensors* **2019**, *19*, 1305. [CrossRef]
4. Tognon, M.; Chávez, H.A.T.; Gasparin, E.; Sablé, Q.; Bicego, D.; Mallet, A.; Lany, M.; Santi, G.; Revaz, B.; Cortés, J.; et al. A truly-redundant aerial manipulator system with application to push-and-slide inspection in industrial plants. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1846–1851. [CrossRef]
5. Park, S.; Lee, J.; Ahn, J.; Kim, M.; Her, J.; Yang, G.H.; Lee, D. Odar: Aerial manipulation platform enabling omnidirectional wrench generation. *IEEE/ASME Trans. Mechatron.* **2018**, *23*, 1907–1918. [CrossRef]
6. Paul, H.; Ono, K.; Ladig, R.; Shimonomura, K. A Multirotor Platform Employing a Three-Axis Vertical Articulated Robotic Arm for Aerial Manipulation Tasks. In Proceedings of the 2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Auckland, New Zealand, 9–12 July 2018; pp. 478–485. [CrossRef]
7. Suarez, A.; Sanchez-Cuevas, P.; Fernandez, M.; Perez, M.; Heredia, G.; Ollero, A. Lightweight and Compliant Long Reach Aerial Manipulator for Inspection Operations. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 6746–6752. [CrossRef]
8. Bodie, K.; Brunner, M.; Pantic, M.; Walser, S.; Pfändler, P.; Angst, U.; Siegwart, R.; Nieto, J. Active interaction force control for contact-based inspection with a fully actuated aerial vehicle. *IEEE Trans. Robot.* **2020**, *37*, 709–722. [CrossRef]
9. Nenchev, D.N. Redundancy resolution through local optimization: A review. *J. Robot. Syst.* **1989**, *6*, 769–798. [CrossRef]
10. Cocuzza, S.; Pretto, I.; Debei, S. Novel reaction control techniques for redundant space manipulators: Theory and simulated microgravity tests. *Acta Astronaut.* **2011**, *68*, 1712–1721. [CrossRef]
11. Papadopoulos, E.; Dubowsky, S. On the nature of control algorithms for free-floating space manipulators. *IEEE Trans. Robot. Autom.* **1991**, *7*, 750–758. [CrossRef]
12. Umetani, Y.; Yoshida, K. Resolved motion rate control of space manipulators with generalized Jacobian matrix. *IEEE Trans. Robot. Autom.* **1989**, *5*, 303–314. [CrossRef]
13. Pasetto, A.; Vyas, Y.; Cocuzza, S. Zero Reaction Torque Trajectory Tracking of an Aerial Manipulator through Extended Generalized Jacobian. *Appl. Sci.* **2022**, *12*, 12254. [CrossRef]
14. Huang, P.; Xu, Y.; Liang, B. Dynamic balance control of multi-arm free-floating space robots. *Int. J. Adv. Robot. Syst.* **2005**, *2*, 13. [CrossRef]

15. Clark, A.B.; Baron, N.; Orr, L.; Kovac, M.; Rojas, N. On a Balanced Delta Robot for Precise Aerial Manipulation: Implementation, Testing, and Lessons for Future Designs. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 7359–7366. [CrossRef]
16. Cocuzza, S.; Pretto, I.; Debei, S. Least-squares-based reaction control of space manipulators. *J. Guid. Control. Dyn.* **2012**, *35*, 976–986. [CrossRef]
17. Cocuzza, S.; Pretto, I.; Debei, S. Reaction torque control of redundant space robotic systems for orbital maintenance and simulated microgravity tests. *Acta Astronaut.* **2010**, *67*, 285–295. [CrossRef]
18. Ivanovic, A.; Car, M.; Orsag, M.; Bogdan, S. Exploiting null space in aerial manipulation through model-in-the-loop motion planning. In Proceedings of the 2020 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 1–4 September 2020; pp. 686–693.
19. Danko, T.W.; Oh, P.Y. Design and control of a hyper-redundant manipulator for mobile manipulating unmanned aerial vehicles. *J. Intell. Robot. Syst.* **2014**, *73*, 709–723. [CrossRef]
20. Wilde, M.; Kwok Choon, S.; Grompone, A.; Romano, M. Equations of motion of free-floating spacecraft-manipulator systems: an engineer's tutorial. *Front. Robot. AI* **2018**, *5*, 41. [CrossRef] [PubMed]
21. Rackl, W.; Lampariello, R.; Albu-Schäffer, A. Parameter Identification Methods for Free-Floating Space Robots with direct Torque Sensing. *IFAC Proc. Vol.* **2013**, *46*, 464–469. [CrossRef]
22. Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. *Robotics: Modelling, Planning and Control*, 1st ed.; Springer: Berlin/Heidelberg, Germany, 2008.
23. Caccavale, F.; Siciliano, B. Kinematic control of redundant free-floating robotic systems. *Adv. Robot.* **2001**, *15*, 429–448. [CrossRef]