2015 IEEE International Conference on Robotics and Automation (ICRA)
Washington State Convention Center
Seattle, Washington, May 26-30, 2015

Quadrotor Landing on an Inclined Platform of a Moving Ground Vehicle

Panagiotis Vlantis, Panos Marantos, Charalampos P. Bechlioulis, and Kostas J. Kyriakopoulos

***Abstract*— In this work we study the problem of landing a quadrotor on an inclined moving platform. The aerial robot employs an forward looking on-board camera to detect and observe the landing platform, which is carried by a mobile robot moving independently on an inclined surface. The platform may also be tilted with respect to the mobile robot. The overall goal is to design the aerial robot's control inputs such that it initially approaches the platform, while maintaining it within the camera's field of view and finally lands on it, in a way that minimizes the errors in position, attitude and velocity, while avoiding collision. Owing to the inclined ground and landing surface, the desired final state of the aerial robot is not an equilibrium state, which complicates significantly the control design. In that respect, a discrete-time non-linear model predictive controller was developed that optimizes both the trajectories and the time horizon, towards achieving the aforementioned objectives while respecting the input constraints as well. Finally, an extensive experimental study, with a Pioneer mobile robot and a Parrot ARDrone quadrotor, clarifies and verifies the theoretical findings.**

I. Introduction

During the last decades, several works in the related literature study the problem of vertical takeoff and landing (VToL) of unmanned aerial vehicles (UAVs) on horizontal fixed surfaces [1] [2]. However, as even more practical applications of UAVs are considered, such as landing on moving vehicles for refueling and grasping objects in mid-air, more complicated scenarios need to be addressed requiring thus the design of more sophisticated controllers.

In this work, we study the problem of landing a quadrotor aerial robot on a platform mounted on a moving wheeled mobile robot. The mobile robot follows a trajectory without cooperating with the aerial robot that tries to land on its platform. The aerial robot is equipped with a fixed forward looking on-board camera that tracks the platform and estimates its state as it approaches and initiates subsequently the landing procedures. More difficulties arise since the ground may not be horizontal and the platform may be inclined with respect to the mobile robot. Given the control objective of minimizing the errors in position, attitude and velocities as well as the fact that the landing surface is inclined the final desired state of the system is not an equilibrium state of the quadrotor. Moreover, the input constraints limit the capabilities of the aerial robot. Last but not least, the aforementioned tasks should be performed while avoiding any form of collision between the aerial robot and the platform. In this respect, notice that certain tasks listed above conflict with others (e.g., tracking the platform while landing on it) whereas considering dynamic environments necessitates for online solutions.

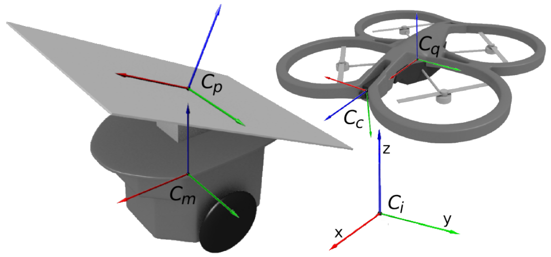

Fig. 1: Earth-fixed and body-fixed frames.

The problem of landing an aerial robot on moving or inclined surfaces has been studied in many recent works. In [3], the vertical landing problem of a quadrotor on a fixed inclined surface was addressed using lasers to estimate the distance from and inclination of the platform, which were used for the generation of a hybrid trajectory. In another recent work [4], a landing maneuver with a quadrotor was executed using a hybrid automaton where the quadrotor collided with and slid up the fixed landing platform before it finally got aligned with it. The most extreme case of perching a quadrotor on vertical surface was demonstrated in [5] as one of the applications of a method for designing trajectories and controllers for aggressive maneuvers tuned through automated experimental trials. Regarding the problem of landing UAVs on top of moving ground vehicles, in [6] a non-linear 2D tracking controller was employed to land a quadrotor on top of a moving horizontal platform by decomposing the problem in initially tracking and finally descending on the platform. Moreover, the coordinated landing of a quadrotor on a ground vehicle is presented in [7], where a joint decentralized controller is designed to accomplish rendezvous and landing.

In this paper, a discrete-time non-linear model predictive controller (MPC) [8] [9] is designed to successfully land a quadrotor on a moving and inclined platform via solving online a computationally intense constrained optimization problem. Contrary to the aforementioned works, the aerial robot's trajectory is designed in real time and is tracked by a single controller, taking at each moment full advantage of the aerial robot's capabilities while also handling the trade-off between the camera, collision avoidance and bounded-input constraints mentioned previously. We also show that the difficulty of reaching a non-equilibrium goal can be tackled via solving for both the trajectories and the controller's time horizon at each iteration.

The outline of the paper is as follows. In Section II we define the models of the two robots and state the control problem. In Section III we formulate the optimization problem that is solved by the MPC in order to land the aerial robot and describe in detail the cost function and the constraints. Finally, in Section IV we describe the setup and present the experimental results.

The authors are with the Control System Lab, School of Mechanical Engineering, National Technical University of Athens, Greece. panagiotis.vlantis.mc@gmail.com, {marantos, chmpechl, kkyria}@mail.ntua.gr

978-1-4799-6923-4/15/$31.00 ©2015 IEEE

II. Problem formulation

Let C_i be the earth-fixed frame and C_q, C_c, C_m, C_p be the body-fixed frames of the aerial robot, the camera, the mobile robot and the landing platform respectively as seen in Fig. 1. Throughout this paper, the notation ${}^{j}_{i}\boldsymbol{R} \in SO^3$ denotes the rotation matrix defining the orientation of frame C_j with respect to frame C_i, ${}^{k}_{i}\boldsymbol{T}_{\boldsymbol{j}} \in \mathbb{R}^{3\times1}$ denotes the offset of frame C_j with respect to C_i, expressed in C_k, $diag(\mathtt{X},\mathtt{Y},\mathtt{Z}) \in \mathbb{R}^{3\times3}$ denotes a diagonal matrix with entries $\mathtt{X},\mathtt{Y},\mathtt{Z}$, and $\mathtt{X}_{(\mathtt{k})}$ stands for the value of symbol X at the end of the k-th time step. Also, the commonly used abbreviations $c\alpha = cos(\alpha)$ and $s\alpha = sin(\alpha)$ for trigonometric functions are employed. Finally, $\boldsymbol{R}(\boldsymbol{\phi})$ and $\boldsymbol{E}(\boldsymbol{\phi})$ stand for the rotation and lumped transformation matrices in terms of the Euler ZYX angles $\boldsymbol{\phi} = [\phi, \theta, \psi]^T$.

A. Aerial Robot and Camera

The hybrid kinematic and dynamic model of the quadrotor, assuming that C_q is placed at the quadrotor's center of mass and following [10], is given by:

$$\dot{\boldsymbol{P}}_{qe} = \boldsymbol{V}_{qe} \tag{1a}$$

$$\dot{\boldsymbol{\phi}}_q = \boldsymbol{E}^{-1}(\boldsymbol{\phi}_q) \cdot \boldsymbol{\omega}_{qb} \tag{1b}$$

$$\dot{\boldsymbol{V}}_{qe} = m^{-1} \cdot {}^{i}_{q}\boldsymbol{R} \cdot (\boldsymbol{F} + \boldsymbol{F_D}) + \boldsymbol{g} \tag{1c}$$

$$\dot{\boldsymbol{\omega}}_{qb} = \boldsymbol{I}^{-1} \cdot (\boldsymbol{\tau} + \boldsymbol{\tau_D} - \boldsymbol{\omega}_{qb} \times (\boldsymbol{I} \cdot \boldsymbol{\omega}_{qb})) \tag{1d}$$

where $\boldsymbol{P}_{qe} = [x_q, y_q, z_q]^T$ and $\boldsymbol{V}_{qe} = [\dot{x}_q, \dot{y}_q, \dot{z}_q]^T$ are the aerial robot's position and linear velocity expressed in C_i, $\boldsymbol{\phi}_q = [\phi_q, \theta_q, \psi_q]^T$ is the vector of Euler ZYX angles describing its attitude relative to C_i, $\boldsymbol{\omega}_{qb} = [p_q, q_q, r_q]^T$ is its angular velocity expressed in C_q, ${}^{i}_{q}\boldsymbol{R} = \boldsymbol{R}(\boldsymbol{\phi}_q)$, and $\boldsymbol{F} = [0, 0, F_z]^T$, $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]$ are the force and torque vectors of the rotors. The constants m and $\boldsymbol{I} = diag(I_{xx}, I_{yy}, I_{zz})$ are the mass and inertia matrix of the aerial robot with I_{xx}, I_{yy}, I_{zz} denoting the principal moments of inertia, and $\boldsymbol{g} = [0, 0, g]^T$ is the gravity vector with $g = -9.81 m/s^2$. The aerodynamic drag force $\boldsymbol{F_D} = -diag(d_u, d_v, d_w) \cdot {}^{i}_{q}\boldsymbol{R}^T \cdot \boldsymbol{V}_{qe}$ and torque $\boldsymbol{\tau_D} = -diag(d_p, d_q, d_r) \cdot \boldsymbol{\omega}_{qb}$ were also incorporated in the model, where d_u, d_v, d_w, d_p, d_q, d_r are positive constant drag coefficients.

The force and torque vectors produced by the four rotors are calculated by an on-board low-level controller via:

$$F_z = \frac{m}{c\theta_q \cdot c\phi_q} \cdot (K_{\dot{z}} \cdot (\dot{z}_{des} - \dot{z}_q) - g) \tag{2a}$$

$$\tau_x = I_{xx} \cdot (K_\phi \cdot (\phi_{des} - \phi_q) - K_p \cdot p_q) - (I_{yy} - I_{zz}) \cdot q_q \cdot r_q \tag{2b}$$

$$\tau_y = I_{yy} \cdot (K_\theta \cdot (\theta_{des} - \theta_q) - K_q \cdot q_q) - (I_{zz} - I_{xx}) \cdot p_q \cdot r_q \tag{2c}$$

$$\tau_z = I_{zz} \cdot K_r \cdot (r_{des} - r_q) - (I_{xx} - I_{yy}) \cdot p_q \cdot q_q \tag{2d}$$

where $\dot{z}_{des}, \phi_{des}, \theta_{des}, r_{des}$ are the commanded values for the linear velocity $\dot{z}_q$, the angles ϕ_q and θ_q, and the angular rate r_q, and $K_{\dot{z}}, K_p, K_\phi, K_q, K_\theta, K_r$ are the internal control gains. Moreover, the input vector $\boldsymbol{u}_q = [\dot{z}_{des}, \phi_{des}, \theta_{des}, r_{des}]^T$ should satisfy the box constraint:

$$-\boldsymbol{u}_B \leq \boldsymbol{u}_q \leq +\boldsymbol{u}_B \tag{3}$$

where $\boldsymbol{u}_B = [\dot{z}_B, \phi_B, \theta_B, r_B]$ are the input bounds imposed by the low-level controller, with $\dot{z}_B$, ϕ_B, θ_B, r_B positive constants.

Hence, substituting (2) in (1) and approximating the left hand side of (1) with the corresponding forward finite differences, yields the discrete-time model of the aerial robot which is defined as follows:

$$\boldsymbol{x}_{q(k+1)} = \boldsymbol{f}_q(\boldsymbol{x}_{q(k)}, \boldsymbol{u}_{q(k)}) \tag{4}$$

where $\boldsymbol{x}_q = [\boldsymbol{P}^T_{qe}, \boldsymbol{V}^T_{qe}, \boldsymbol{\phi}^T_q, \boldsymbol{\omega}^T_{qb}]^T$ is the state vector and $\boldsymbol{f}_q(\cdot, \cdot) : \mathbb{R}^{12} \times \mathbb{R}^4 \mapsto \mathbb{R}^{12}$ is an almost everywhere continuously differentiable function, except from $\phi_q = 2 \cdot \pi \cdot l_1 + \frac{\pi}{2}$ and $\theta_q = 2 \cdot \pi \cdot l_2 + \frac{\pi}{2}$, $\forall l_1, l_2 \in \mathbb{N}$.

The aerial robot is also equipped with an on-board camera facing forwards, with its frame C_c placed at the camera's center. Its z-axis is normal to the camera's plane. The position $\boldsymbol{P}_{ce}$ of the camera, expressed in C_i, and the corresponding rotation matrix ${}^{i}_{c}\boldsymbol{R}$ are obtained in terms of the quadrotor's state according to $\boldsymbol{P}_{ce} = \boldsymbol{P}_{qe} + {}^{i}_{q}\boldsymbol{R} \cdot {}^{q}_{q}\boldsymbol{T}_{\boldsymbol{c}}$ and ${}^{i}_{c}\boldsymbol{R} = {}^{i}_{q}\boldsymbol{R} \cdot {}^{q}_{c}\boldsymbol{R}$, with ${}^{q}_{q}\boldsymbol{T}_{\boldsymbol{c}}$ and ${}^{q}_{c}\boldsymbol{R}$ being the fixed offset and fixed rotation matrix of C_q with respect to C_c.

B. Mobile Robot and Platform

The ground vehicle is modeled as a differential wheeled mobile robot, the kinematics of which is given by:

$$\dot{\boldsymbol{P}}_{me} = \boldsymbol{V}_{me} \tag{5a}$$

$$\dot{\boldsymbol{\phi}}_m = \boldsymbol{E}^{-1}(\boldsymbol{\phi}_m) \cdot \boldsymbol{\omega}_{mb} \tag{5b}$$

where, similarly to the quadrotor, $\boldsymbol{P}_{me} = [x_m, y_m, z_m]^T$ and $\boldsymbol{V}_{me} = [\dot{x}_m, \dot{y}_m, \dot{z}_m]^T$ denote the mobile robot's position and linear velocity expressed in C_i, $\boldsymbol{\phi}_m = [\phi_m, \theta_m, \psi_m]^T$ the vector of its Euler ZYX angles, and $\boldsymbol{\omega}_{mb} = [p_m, q_m, r_m]^T$ its angular velocity expressed in C_m, An on-board velocity controller is used to achieve the commanded velocities v_{des} and ω_{des} calculated by:

$$\boldsymbol{V}_{me} = {}^{i}_{m}\boldsymbol{R} \cdot \boldsymbol{V}_{mbc} \tag{6a}$$

$$\boldsymbol{\omega}_{mb} = \boldsymbol{\omega}_{mbc} \tag{6b}$$

where $\boldsymbol{V}_{mbc} = [v_{des}, 0, 0]^T$ and $\boldsymbol{\omega}_{mbc} = [0, 0, \omega_{des}]^T$ are the linear and angular body velocities of the mobile robot and ${}^{i}_{m}\boldsymbol{R} = \boldsymbol{R}(\boldsymbol{\phi}_m)$.

Thus, substituting (6) in (5) and using forward finite differences yields the corresponding discrete-time model of the mobile robot, defined as:

$$\boldsymbol{x}_{m(k+1)} = \boldsymbol{f}_m(\boldsymbol{x}_{m(k)}, \boldsymbol{u}_{m(k)}) \tag{7}$$

where $\boldsymbol{x}_m = [\boldsymbol{P}^T_{me}, \boldsymbol{V}^T_{me}, \boldsymbol{\phi}^T_m, \boldsymbol{\omega}^T_{mb}]^T$ is the state vector, $\boldsymbol{u}_m = [v_{des}, \omega_{des}]^T$ is the input vector, and $\boldsymbol{f}_m : \mathbb{R}^{12} \times \mathbb{R}^2 \mapsto \mathbb{R}^{12}$ assuming $\theta_m \neq 2 \cdot \pi \cdot l + \frac{\pi}{2}$, $\forall l \in \mathbb{N}$.

Additionally, an inclined rectangular flat platform is mounted on the top of the mobile robot with the corresponding frame C_p located at its center and the z-axis pointing outwards perpendicularly to the plane it defines. Its corresponding position $\boldsymbol{P}_{pe}$ and velocities $\boldsymbol{V}_{pe}$ and $\boldsymbol{\omega}_{pb}$ can be obtained in terms of the mobile's state by:

$$\boldsymbol{P}_{pe} = \boldsymbol{P}_{me} + {}^{i}_{m}\boldsymbol{R} \cdot {}^{m}_{m}\boldsymbol{T}_{\boldsymbol{p}} \tag{8a}$$

$$\boldsymbol{V}_{pe} = \boldsymbol{V}_{me} + {}^{i}_{m}\boldsymbol{R} \cdot (\boldsymbol{\omega}_{mb} \times {}^{m}_{m}\boldsymbol{T}_{\boldsymbol{p}}) \tag{8b}$$

$$\boldsymbol{\omega}_{pb} = {}^{m}_{p}\boldsymbol{R}^T \cdot \boldsymbol{\omega}_{mb} \tag{8c}$$

where ${}^{m}_{m}\boldsymbol{T}_{\boldsymbol{p}}$ and ${}^{m}_{p}\boldsymbol{R}$ are fixed. The rotation matrix ${}^{i}_{p}\boldsymbol{R}$ relating frames C_i and C_p can be easily calculated by ${}^{i}_{p}\boldsymbol{R} = {}^{i}_{m}\boldsymbol{R} \cdot {}^{m}_{p}\boldsymbol{R}$.

C. Control Problem

We consider a scenario where the mobile robot is tracking a trajectory, unknown to the aerial robot, and the platform is initially visible by its camera. The control objective is to design a high-level controller that calculates the aerial robot's control inputs $\boldsymbol{u}_q$ such that it approaches the platform and executes a landing maneuver minimizing the errors in position, attitude and velocities between the aerial robot and the platform. Given that the platform is inclined, the goal is not an equilibrium state of the quadrotor. Moreover, during the approaching phase, visual contact with the platform should not be compromised. Furthermore, the whole trajectory should be executed without the aerial robot penetrating the plane defined by the platform since otherwise unwanted situations such as violation of the visual constraints or collision with the landing surface would occur. Finally, bounds on the quadrotor's inputs imposed by the on-board low-level controller should also be taken into account.

III. Controller design

A constrained discrete-time non-linear MPC is designed to solve the aforementioned control problem. At the beginning of each iteration, (i.e., the time step $k = 0$), the MPC utilizes the aerial robot's current state estimation $\boldsymbol{x}_{q_o}$, the last command $\boldsymbol{u}_{q_o}$ sent to the aerial robot and the mobile robot's estimated initial state and input vectors $\boldsymbol{x}_{mo}$ and $\boldsymbol{u}_{mo}$, whereas at the end of it returns the optimal trajectories X_{opt} and U_{opt} of the aerial robot and the optimal time horizon K_{opt}. In order to achieve the non-equilibrium goal, integral costs alone are not enough as they cannot guarantee that all states converge to the desired values at the same instant. On the other hand, the presence of terminal costs yields trajectories such that the goal state could be reached at the end of a sufficiently large horizon. Therefore, assuming that the length of the horizon is appropriately updated at each iteration, minimization of a weighted sum of integral and terminal costs could take the aerial robot to the desired non-equilibrium state at the end of the maneuver, thus the necessity of solving for K_{opt}.

Moreover, since the mobile robot is moving independently, the prediction of its unknown trajectory is needed in the MPC in order to accurately plan the aerial robot's actions, which is obtained before the optimization begins, by propagating the mobile robot's model as defined in (7) assuming constant commanded body velocities $\boldsymbol{u}_{m(k)} = \boldsymbol{u}_{mo}$ during the entire prediction. Furthermore, considering that the duration of each iteration is equal to the fixed time interval Δt of each time step k, $\forall k = 0, 1, \ldots, K$, the next command sent to the aerial robot should be $\boldsymbol{u}_{q(1)}$ instead of $\boldsymbol{u}_{q(0)}$, which should be removed from the optimization's set of free variables.

A. Optimization Problem

The optimization problem that is solved by the MPC at each iteration is stated as follows:

$$\min_{K} \min_{\boldsymbol{X},\boldsymbol{U}} J(\boldsymbol{X},\boldsymbol{U},K) \tag{9a}$$

$$J(\boldsymbol{X},\boldsymbol{U},K) = J_L(\boldsymbol{X},K) + J_A(\boldsymbol{X},\boldsymbol{U},K) + J_V(\boldsymbol{X},K) \tag{9b}$$

$$\boldsymbol{x}_{q(k+1)} = \boldsymbol{f}_q(\boldsymbol{x}_{q(k)}, \boldsymbol{u}_{q(k)}) \qquad k = 0, 1, \ldots, K{-}1 \tag{9c}$$

$$\boldsymbol{x}_{q(0)} = \boldsymbol{x}_{q_o} \qquad \boldsymbol{u}_{q(0)} = \boldsymbol{u}_{q_o} \tag{9d}$$

$$g_c(\boldsymbol{x}_{q(k)}) < 0 \qquad k = 1, 2, \ldots, K \tag{9e}$$

$$g_b(\boldsymbol{u}_{q(k)}) < 0 \qquad k = 0, 1, \ldots, K{-}1 \tag{9f}$$

$$0 \leq K \leq K_{max} \qquad K \in \mathbb{N} \tag{9g}$$

where $\boldsymbol{X} = [\boldsymbol{x}_{q\,(0)}^T, \boldsymbol{x}_{q\,(1)}^T, ..., \boldsymbol{x}_{q\,(K)}^T]^T$ is the state trajectory and $\boldsymbol{U} = [\boldsymbol{u}_{q\,(0)}^T, \boldsymbol{u}_{q\,(1)}^T, ..., \boldsymbol{u}_{q\,(K-1)}^T]^T$ is the input trajectory of the aerial robot, for a given time horizon K. It consists of two nested constrained minimization sub-problems, the inner being solved for $\boldsymbol{X}$ and $\boldsymbol{U}$ and the outer for K. The cost function J is composed of three terms denoting the tasks of:

i. taking the aerial robot close to the platform (approaching phase cost J_A),
ii. keeping the center of the platform within the camera's field of view during the approaching phase (visual constraint cost J_V), and
iii. landing the aerial robot on the platform with aligned frames (landing cost J_L).

Given $\boldsymbol{x}_{q_o}$ and $\boldsymbol{u}_{q_o}$, the quadrotor's model, as defined in (4), is used to propagate the aerial robot's state. The collision avoidance and the bounds imposed on the control inputs are modeled as inequality constraints, where $g_c : \mathbb{R}^{12} \mapsto \mathbb{R}$ and $g_b : \mathbb{R}^4 \mapsto \mathbb{R}$ are continuously differentiable functions. Furthermore, the horizon K is bounded from above by a sufficiently large K_{max} to ensure that the aerial robot can perform the landing maneuver within the specified amount of time steps.

B. Cost Function

The term J_A, that corresponds to the approaching phase, is composed of the following two quadratic terms:

$$J_A = w_a \cdot \Big(\sum_{k=0}^{K} \boldsymbol{x}_{A(k)}^T \cdot \boldsymbol{Q}_A \cdot \boldsymbol{x}_{A(k)} + \sum_{k=0}^{K-2} \Delta \boldsymbol{u}_{q\,(k)}^T \cdot \boldsymbol{R}_A \cdot \Delta \boldsymbol{u}_{q(k)} \Big) \tag{10a}$$

$$w_a = \frac{1}{K+1} \cdot \left(\delta + (1{-}\delta) \cdot \left(\frac{d_{init}}{d_{a,ref} {+} d_{init}} \right)^2 \right) \tag{10b}$$

where $\boldsymbol{x}_A = [(\boldsymbol{P}_{qe} - \boldsymbol{P}_{pe})^T, \boldsymbol{V}_{qe}, \boldsymbol{\phi}_q, \boldsymbol{\omega}_{qb}]^T$, $\Delta \boldsymbol{u}_{q(k)} = \boldsymbol{u}_{q(k+1)} - \boldsymbol{u}_{q(k)}$, $\boldsymbol{Q}_A \in \mathbb{R}^{12 \times 12}$ and $\boldsymbol{R}_A \in \mathbb{R}^{4 \times 4}$. The constant weight $\boldsymbol{Q}_A > 0$ of the first term affects the decreasing rate of the distance between the two robots, while the weight $\boldsymbol{R}_A \geq 0$ of the second determines the smoothness of the control inputs trajectory. The term w_a is a strictly increasing function of the initial distance $d_{init} = \|\boldsymbol{P}_{qe(0)} - \boldsymbol{P}_{me(0)}\|$, mapping $[0, +\infty)$ to $[\delta, 1]$, used to reduce the contribution of J_A as the aerial robot approaches the platform, for a certain constant $\delta \in (0, 1)$ and reference distance $d_{a,ref} > 0$. Notice also that both sums are divided by $K{+}1$ in order to obtain their corresponding mean values over time.

Regarding the visual constraint penalty J_V, given the camera's horizontal and vertical angles of view, a_h and a_v, an elliptic cone with its apex placed at the origin of C_c and its z-axis aligned with the camera's z-axis, is used to approximate the visible space, as it results in more conservative tracking of the platform. The equation of the cone, multiplied by the squared distance between the camera and the platform, is used to penalize the distance between the platform's center and the cone's axis as follows:

$$J_V = w_v \cdot \sum_{k=0}^{K} d_{pc(k)}^2 \cdot \bigg(\Big(\frac{x_{pc(k)}}{\alpha \cdot \left| z_{pc(k)} \right| {+} \varepsilon} \Big)^2 {+} \Big(\frac{y_{pc(k)}}{\beta \cdot \left| z_{pc(k)} \right| {+} \varepsilon} \Big)^2 \bigg) \tag{11a}$$

$$w_v = \frac{W_V}{K+1} \cdot \left(\frac{d_{init}}{d_{v,ref} + d_{init}} \right)^2 \tag{11b}$$

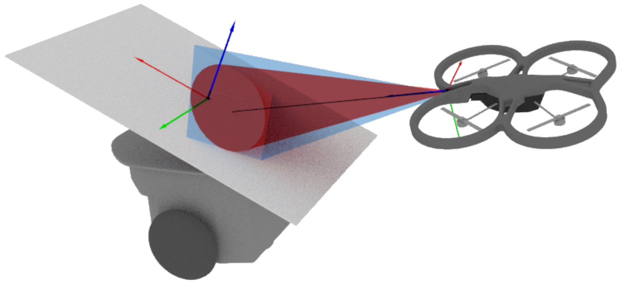

Fig. 2: Camera's actual (blue pyramid) and approximate (red cone) fields of view.

where $\boldsymbol{P}_{pc} = {}^{\boldsymbol{i}}_{\boldsymbol{c}}\boldsymbol{R}^T \cdot (\boldsymbol{P}_{pe} - \boldsymbol{P}_{ce}) = [x_{pc}, y_{pc}, z_{pc}]^T$ is the position of the platform relative to the camera, expressed in the C_c, $d_{pc} = \|\boldsymbol{P}_{pc}\|$ is the distance between the camera and the platform, $\alpha = tan(\frac{a_h}{2})$ and $\beta = tan(\frac{a_v}{2})$, and $W_V > 0$ is the corresponding constant weight of the visual constraint's cost. An additional small constant $\varepsilon > 0$ was added in order to deal with the singularity the cone has at its origin. Similarly, the sum of the individual costs is multiplied by w_v which goes to zero as the aerial robot approaches the target, given some constant reference distance $d_{v,ref} > 0$.

The final cost corresponding to the landing of the aerial robot are given by:

$$J_L = W_L \cdot K^2 + \boldsymbol{x}_{L}{}^{T}_{(K)} \cdot \boldsymbol{Q}_L \cdot \boldsymbol{x}_{L(K)} \tag{12}$$

where $\boldsymbol{x}_L = [(\boldsymbol{P}_{qe} - \boldsymbol{P}_{pe})^T_, (\boldsymbol{V}_{qe} - \boldsymbol{V}_{pe})^T_, \boldsymbol{\gamma}^T_, (\boldsymbol{\omega}_{qb} - {}^{\boldsymbol{q}}_{\boldsymbol{p}}\boldsymbol{R} \cdot \boldsymbol{\omega}_{pb})]^T$, $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \gamma_3]^T$ are the angles between the pairs of x, y, z axes of frames C_q and C_p respectively calculated by $\gamma_i = acos(\boldsymbol{e}_i^T \cdot {}^{\boldsymbol{i}}_{\boldsymbol{q}}\boldsymbol{R}^T \cdot {}^{\boldsymbol{i}}_{\boldsymbol{p}}\boldsymbol{R} \cdot \boldsymbol{e}_i)$, $\forall i \in \{1, 2, 3\}$ with $\boldsymbol{e}_i$ being the unit vector whose i-th element is 1, ${}^{\boldsymbol{q}}_{\boldsymbol{p}}\boldsymbol{R} = {}^{\boldsymbol{i}}_{\boldsymbol{q}}\boldsymbol{R}^T \cdot {}^{\boldsymbol{p}}_{\boldsymbol{i}}\boldsymbol{R}$, and $W_L, \boldsymbol{Q}_L > 0$ are corresponding constant weights, with $\boldsymbol{Q}_L \in \mathbb{R}^{12 \times 12}$. The first of the two terms is used to affect the duration of the maneuver while the second determines how close the final position, attitude and velocities of the aerial robot will be to the corresponding values of the platform.

C. Collision Avoidance and Bounded Input Constraints

The collision avoidance problem between the aerial robot and the platform is addressed by detecting the intersection between a bounding volume, enclosing the aerial robot, and the infinite plane defined by the flat rectangular landing platform. The following superellipsoid:

$$\left(\frac{x}{a}\right)^{2 \cdot \nu} + \left(\frac{y}{b}\right)^{2 \cdot \nu} + \left(\frac{z}{c}\right)^{2 \cdot \nu} - 1 = 0 \ , \qquad \nu \in \mathbb{N} \tag{13}$$

where x, y, z are expressed in C_q and a, b, c are radii corresponding to the length, width and height of the aerial robot, was chosen as the enclosing surface since it approximates the quadrotor's geometry well (see Fig. 3), for appropriate ν, and requires only one simple test $g_c(\boldsymbol{x}_q) = \lambda_{qp} < 0$ per time step k involving the calculation of the signed distance λ_{qp} between the plane and the superellipsoid (see the Appendix for more details).

Similarly, the inputs constraint (3) was approximated by the following inequality:

$$g_b(\boldsymbol{u}_q) = \left(\frac{\dot{z}_{des}}{\dot{z}_B}\right)^{2 \cdot \mu} + \left(\frac{\phi_{des}}{\phi_B}\right)^{2 \cdot \mu} + \left(\frac{\theta_{des}}{\theta_B}\right)^{2 \cdot \mu} + \left(\frac{r_{des}}{r_B}\right)^{2 \cdot \mu} - 1 < 0 \ , \quad \mu \in \mathbb{N} \tag{14}$$

for sufficiently large μ, since only one instead of eight inequalities per step is added to the problem, reducing thus significantly the size of the constraints.

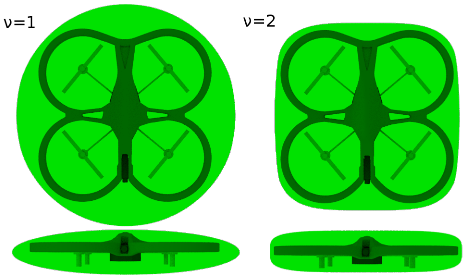

Fig. 3: Superellipsoid used as a bounding volume approximating the quadrotor's shape, for different values of ν.

D. Solver

The algorithm employed for solving the optimization problem (9) is given as follows:

1. Set $K_{mean} = K_{init}$, where K_{init} is the initial estimation of the optimal time horizon K_{opt}.
2. For each feasible $K_i = K_{mean} + i$ satisfying (9g), solve the corresponding inner minimization problem for the optimal trajectories $\boldsymbol{X}_i$ and $\boldsymbol{U}_i$, yielding cost $J_i(K_i, \boldsymbol{X}_i, \boldsymbol{U}_i)$, under the constraints (9b)- (9f), $\forall i \in \mathbb{K} = \{-r, -r{+}1, \ldots, r\}$ with $r \in \mathbb{N}_{>0}$.
3. Find the iteration's overall optimal solution (K_{opt}, $\boldsymbol{X}_{opt}$, $\boldsymbol{U}_{opt}$, J_{opt}) = (K_j, $\boldsymbol{X}_j$, $\boldsymbol{U}_j$, J_j), with $j = argmin_{i \in \mathbb{K}}(J_i)$.
4. If $K_{opt} = 0$, the maneuver is complete, otherwise set $K_{mean} = K_{opt}$ and go back to step 2.

The solution of the inner minimization problems is based on the LBFGS and Quadratic Interpolation algorithms after converting the original constrained optimization problems to unconstrained ones via the Augmented Lagrange Multipliers method. It should also be noticed that the aforementioned gradient-based methods require that the cost function and the constraints are continuously differentiable functions of the state and inputs trajectories, which holds in our case as long as the input bounds $\boldsymbol{u}_B$ and internal control gains are such that $|\phi_{q(k)}| < \frac{\pi}{2}$ and $|\theta_{q(k)}| < \frac{\pi}{2}$, $\forall k \in \{0, 1, \ldots, K\}$.

IV. Experimental results

A. Setup

A Parrot ARDrone2 quadrotor and a Pioneer P2-DX mobile robot were used during the experiments. A cardboard with dimensions $100cm \times 70cm$ was mounted on the ground vehicle with a marker located at its center [11]. The quadrotor is equipped with an on-board computer, an inertia measurement unit (IMU), an altimeter, a barometer and two cameras; one high-definition facing forwards used for tracking the platform and one low-definition facing downwards for optical flow. The on-board computer runs the internal low-level controller presented previously and estimates some of the quadrotor's states through sensor fusion. The model of the quadrotor was fitted with flight data collected for offline identification in order to determine the unknown parameters of the system.

A communication is established over wifi between the aerial robot and a ground station, where the state estimation of the two robots and the execution of the MPC take place. Two Unscented Kalman Filters (UKF) [12] [13], using only the measurements from the quadrotor's sensors, are employed to estimate the full state vectors of the two robots. The estimated states of the two robots are then passed to the

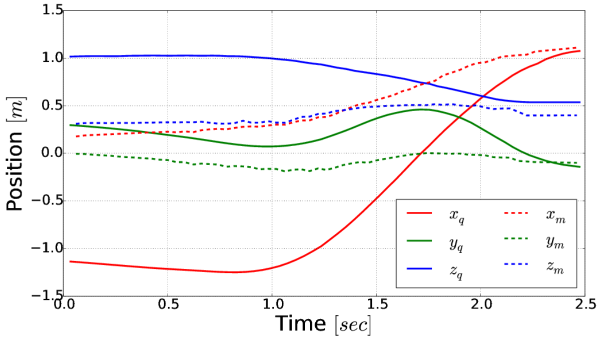

Fig. 4: Estimated positions of the aerial and mobile robots.

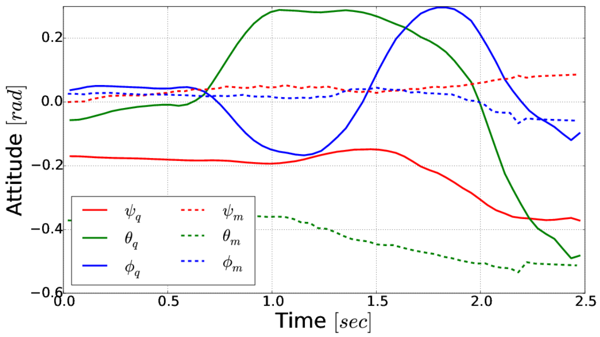

Fig. 5: Estimated attitude of the aerial and mobile robots.

MPC which calculates the commands to be sent to the aerial robot and turns the aerial robot's motors off upon successful completion of the landing maneuver ($K_{opt} = 0$).

B. Results

In this subsection, the results from an outdoor experiment are presented for evaluating the performance of the proposed MPC. The inclination of the platform relative to the mobile robot was $18.5°$ and the inclination of the ground was approximately $13°$. The controller initiates at time $t = 0$, while the quadrotor is in free flight approximately $1.5m$ behind the mobile robot that tracks a given trajectory, and requires approximately $2.5sec$ to successfully complete the landing. The time horizon's upper bound and the controller's sampling period are $K_{max} = 10$ and $\Delta t = 0.08sec$, such that the optimization is completed within the specified interval of each iteration. The bounds of the low-level controller are set to $\dot{z}_B = 1m/sec$, $\phi_B = \theta_B = 0.35rad$, and $r_B = \frac{\pi}{2}rad/sec$. The marker placed on the platform is initially visible by the quadrotor's camera. The visual contact is lost only during the final steps of the maneuver. During that time, the mobile robot's state is updated solely via propagating the model. Furthermore, for the short time the quadrotor is moving right above the platform, the measurements from the altimeter and the results of the optical flow are not taken into consideration in the quadrotor's state estimation in order to avoid the confusion caused by the transition from absolute to relative measurements.

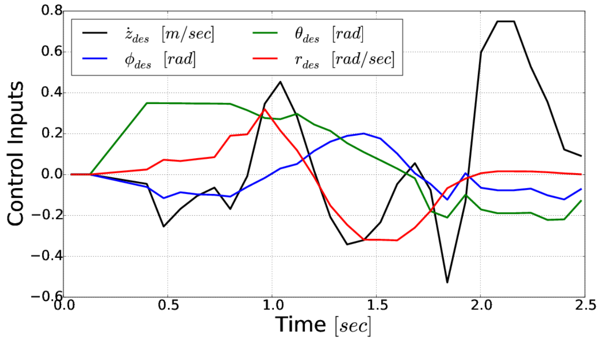

Fig. 6: Aerial robot's control inputs.

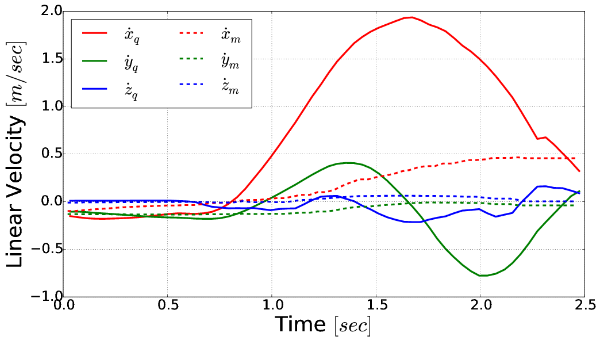

Fig. 7: Estimated linear velocities of the aerial and mobile robots.

Notice in Fig. 4 and Fig. 5 that the errors in the estimated position and attitude go to zero at the end of the maneuver, even though a significant error of approximately $30°$ remains in the yaw angles ψ_q and ψ_m, in order to maintain visibility of the platform as the quadrotor drifted to the left since the controller was not designed to handle external disturbances like wind gusts. Moreover, owing to (13), for $\nu = 2$, $a = b = 0.25m$ and $c = 0.075m$, no collision between the aerial robot and the platform takes place. In Fig. 6, we can see that the quadrotor's inputs are successfully kept within the specified limits as given in (14) with $\mu = 8$. The only time θ_{des} gets close to θ_B is at the beginning of the maneuver, where the quadrotor needs to accelerate in order to reduce the distance between itself and the platform. Also, notice that a final $\theta_q \approx 30°$ could successfully be achieved despite the bound $\theta_B = 20°$ of the corresponding input θ_{des}, owing the MPC taking advantage of both dynamics of the quadrotor's joint model and the low-level controller.

Fig. 7 and Fig. 8 present the linear and angular velocities of the aerial robot and the platform, the error of which, as we can see, goes successfully to zero. Finally, Fig. 9 illustrates how the optimal time horizon K_{opt} changes over time. Initially, as the quadrotor approaches the platform the time horizon is fixed at K_{max} until the distance becomes small enough to complete the maneuver within K_{opt} steps exactly. After that point, the optimal time horizon decreases steadily, by approximately one step per iteration, as one would expect, until the goal state is achieved.

V. Conclusion and Future Work

In this paper, an MPC controller with the goal of landing an quadrotor aerial robot on a moving inclined platform was designed. Visual constraints due to the detection of the platform using an on-board camera, collision avoidance between the platform and the quadrotor and input bounds were also taken into consideration in the solution. We also showed that the deficiency of conventional MPCs to handle

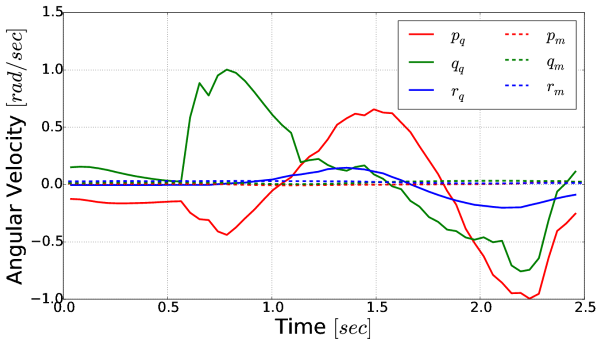

Fig. 8: Estimated angular velocities of the aerial and mobile robots (expressed in C_q).

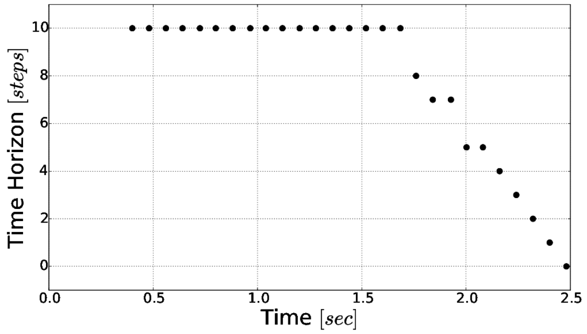

Fig. 9: Optimal time horizon K_{opt}.

non-equilibrium goal states can be overcome by solving for the optimal time horizon along with the trajectories at each iteration. Finally, we presented experimental results demonstrating the efficiency of the proposed controller on a outdoor environment, which can also be seen in the accompanying video. Future research efforts will be devoted towards extending the results to a helicopter's landing process where the collision avoidance with the platform is vital.

Appendix

Collision detection between plane and superellipsoid

Let $\mathbb{Q}$ and $\mathbb{P}$ be a superellipsoid and an infinite plane in $\mathbb{R}^3$. Let C_w be the global coordinate system and C_q, C_p the coordinate systems attached to $\mathbb{Q}$ and $\mathbb{P}$ respectively such that C_q has its origin at the center of $\mathbb{Q}$ and is aligned with the superellipsoid's axes, and C_p has its origin lying on $\mathbb{P}$ and its z-axis is perpendicular to $\mathbb{P}$. The equation defining the surface $\mathbb{Q}$ is given by:

$$f({}^q\boldsymbol{P}) := \left(\frac{{}^qx}{a}\right)^n + \left(\frac{{}^qy}{b}\right)^n + \left(\frac{{}^qz}{c}\right)^n - 1 = 0 \qquad (15)$$

for all ${}^q\boldsymbol{P} \in \mathbb{Q}$, where ${}^q\boldsymbol{P} = [{}^qx, {}^qy, {}^qz]^T$ is expressed in C_q, the constants a, b, c are the radii on x, y, z axes respectively, and $n = 2 \cdot l$ with $l \in \mathbb{N}$.

The corresponding equation of $\mathbb{P}$ can be obtained by transforming a point ${}^p\boldsymbol{P} = [{}^px, {}^py, 0]^T \in \mathbb{P}$ from C_p to C_q, yielding:

$$g({}^q\boldsymbol{P}) := A\cdot{}^qx + B\cdot{}^qy + C\cdot{}^qz + D = 0 \qquad (16)$$

where $[A, B, C, D] = ({}^{\boldsymbol{w}}_{\boldsymbol{p}}\boldsymbol{R}\cdot\boldsymbol{e}_3)^T\cdot[{}^{\boldsymbol{w}}_{\boldsymbol{q}}\boldsymbol{R}, {}^{\boldsymbol{w}}_{\boldsymbol{p}}\boldsymbol{T}_{\boldsymbol{q}}]$ with ${}^{\boldsymbol{w}}_{\boldsymbol{q}}\boldsymbol{R}$ and ${}^{\boldsymbol{w}}_{\boldsymbol{p}}\boldsymbol{R}$ denoting the rotation matrices that express the orientations of C_q and C_p relative to C_w, ${}^{\boldsymbol{w}}_{\boldsymbol{p}}\boldsymbol{T}_{\boldsymbol{q}}$ denoting the offset of C_q relative to C_p, expressed in C_w, and $\boldsymbol{e}_3 = [0, 0, 1]^T$.

Assuming that $\mathbb{Q}$ and $\mathbb{P}$ do not intersect and the plane's gradient $\boldsymbol{V} = [A, B, C]^T$ is pointing towards the superellipsoid, the two points ${}^q\boldsymbol{P}_Q \in C_q$ and ${}^q\boldsymbol{P}_P \in C_p$ with the shortest distance define a line $\mathbb{L}$ that is normal to both surfaces, since $\mathbb{Q}$ is a convex and smooth surface. As a result,

$$\nabla f({}^q\boldsymbol{P}_Q) = -\nabla g({}^q\boldsymbol{P}_P) \;\Rightarrow\; {}^q\boldsymbol{P}_Q = -\begin{bmatrix}\sqrt[n-1]{A/n}\\ \sqrt[n-1]{B/n}\\ \sqrt[n-1]{C/n}\end{bmatrix} \qquad (17)$$

Hence, without loss of generality, the equation of $\mathbb{L}$ in parametric form is defined in:

$$ {}^q\boldsymbol{P}(\lambda) = -\lambda\cdot\frac{\boldsymbol{V}}{\|\boldsymbol{V}\|} + {}^q\boldsymbol{P}_P \qquad \forall\, {}^q\boldsymbol{P} \in \mathbb{L} \qquad (18)$$

where λ is the line's parameter. Thus, solving (18) for ${}^q\boldsymbol{P}_P$, substituting the result in (16) and setting ${}^q\boldsymbol{P}(\lambda) = {}^q\boldsymbol{P}_Q$, yields:

$$\lambda_{qp} = -\|\boldsymbol{V}\|\cdot\frac{\boldsymbol{V}^T\cdot{}^q\boldsymbol{P}_Q + D}{\boldsymbol{V}^T\cdot\boldsymbol{V}} \qquad (19)$$

where λ_{qp} is the signed distance between $\mathbb{Q}$ and $\mathbb{P}$. Since the gradient of the plane was assumed to point towards the superellipsoid, the following statement holds:

$$\lambda_{qp} < 0 \quad \text{iff } \mathbb{Q} \text{ and } \mathbb{P} \text{ do not intersect} \qquad (20a)$$

$$\lambda_{qp} \geq 0 \quad \text{iff } \mathbb{Q} \text{ and } \mathbb{P} \text{ intersect} \qquad (20b)$$

References

[1] K. Wenzel, A. Masselli, and A. Zell, "Automatic take off, tracking and landing of a miniature uav on a moving carrier vehicle," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 61, no. 1-4, pp. 221–238, 2011.

[2] D. Lee, T. Ryan, and H. Kim, "Autonomous landing of a vtol uav on a moving platform using image-based visual servoing," 2012, pp. 971–976.

[3] J. Dougherty, D. Lee, and T. Lee, "Laser-based guidance of a quadrotor uav for precise landing on an inclined surface," in *American Control Conference (ACC), 2014*, June 2014, pp. 1210–1215.

[4] D. Cabecinhas, R. Cunha, and C. Silvestre, "A robust landing and sliding maneuver controller for a quadrotor vehicle on a sloped incline," in *in Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*, June 2014, pp. 523–528.

[5] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[6] H. Voos and H. Bou-Ammar, "Nonlinear tracking and landing controller for quadrotor aerial robots," in *Control Applications (CCA), 2010 IEEE International Conference on*, Sept 2010, pp. 2136–2141.

[7] J. Daly, Y. Ma, and S. Waslander, "Coordinated landing of a quadrotor on a skid-steered ground vehicle in the presence of time delays," *Autonomous Robots*, 2014.

[8] A. Bemporad and M. Morari, *Robust model predictive control: A survey*. London, U.K.: Springer, 1999.

[9] M. Morari and J. H. Leeb, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, pp. 667–682, 1999.

[10] T. Bresciani, "Modeling, identification and control of a quadrotor helicopter," Master's thesis, Lund University, October 2008.

[11] S. Garrido-Jurado, R. Muoz-Salinas, F. Madrid-Cuevas, and M. Marn-Jimnez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280 – 2292, 2014.

[12] E. Wan and R. Van der Merwe, "The unscented kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, 2000, pp. 153–158.

[13] G. Karras, S. Loizou, and K. Kyriakopoulos, "Towards semi-autonomous operation of under-actuated underwater vehicles: sensor fusion, on-line identification and visual servo control," *Autonomous Robots*, vol. 31, no. 1, pp. 67–86, 2011.