Tree Cavity Inspection Using Aerial Robots

Kelly Steich[1,2], Mina Kamel[2], Paul Beardsley[1], Martin K. Obrist[3], Roland Siegwart[2] and Thibault Lachat[3]

***Abstract*— We present an aerial robotic platform for remote tree cavity inspection, based on a hexacopter Micro-Aerial vehicle (MAV) equipped with a dexterous manipulator. The goal is to make the inspection process safer and more efficient and facilitate data collection about tree cavities, which are important for the conservation of biodiversity in forest ecosystems. This work focuses on two key enabling technologies, namely a vision-based cavity detection system and strategies for high level control of the MAV and manipulator. The results of both simulation and real-world experiments are discussed at the end of the paper and demonstrate the effectiveness of our approach.**

I. INTRODUCTION

Aerial robots are gaining significant attention in the research community thanks to their ability to reach inaccessible areas for humans and the possibility to operate in hazardous environment. Aerial robots have proven great potential in many relevant applications such as infrastructure inspection [1]–[3], surveillance and security [4], assistance in natural disasters [5] and crop monitoring [6]. The range of possible applications is extended further by integrating robotic manipulators into aerial robotic systems which enables them to interact with the environment [7], [8].

In this paper, we propose an aerial robotic platform for tree cavity inspection, based on a hexacopter Micro Aerial Vehicle (MAV) equipped with a dexterous manipulator. Within the scope of this work, tree cavity inspection is defined as the problem of (a) detecting a cavity aperture using a depth sensor, (b) hovering in front of the aperture, (c) inserting a stereo camera mounted on an end-effector on a robot manipulator into the cavity and (d) capturing images of the cavity for offline 3D reconstruction. Figure 1 shows a schematic of such a platform and the inspection process.

Tree cavities are used by a multitude of animal species, including birds, mammals and beetles, for shelter, nesting or larval development and their importance for the conservation of biodiversity in forest ecosystems is widely recognized. However, in managed forests, trees with low economic value, which are often those with cavities or those prone to their formation, are routinely removed. Consequently, their number has dramatically decreased in the last century and many cavity dependent species are currently highly endangered. Many researchers now call for an increased conservation effort regarding tree cavities [9]–[12]. To maximize the efficiency in terms of conservation and minimize the associated economic costs, there is need to better understand tree cavities and the characteristics that make them ecologically valuable. However, large scale studies and detailed data, especially about their genesis and dynamic (from small cavity to big cavity with mold), are rare [9], [11]. Usually, tree cavities are observed from the ground using binoculars, resulting in superficial information, by climbing the tree and manually inspecting the inside of the cavity with an endoscopic camera or by using inspection cameras mounted on a telescopic pole. Drawbacks of the latter two methods are the heaviness and bulkiness of the required equipment (ladder or telescopic pole), the very basic image quality, that cannot be used for any metric observations (e.g., dimension and volume of the cavity) and that many trees are too unstable to be climbed. Tree cavity inspection by means of an aerial robot has the potential to solve all these problems, by being safer, more time-efficient and able to gather highly detailed data.

This work focuses on cavities built by woodpeckers. These are the most common type of cavities in managed forests, since woodpeckers excavate a new breeding cavity every year and such trees are increasingly maintained by forest managers. They have circular or elliptical apertures (usually with a diameter of 3-12 cm) and a deep hollow inside space as shown by the cross section of a tree cavity in Figure 1.

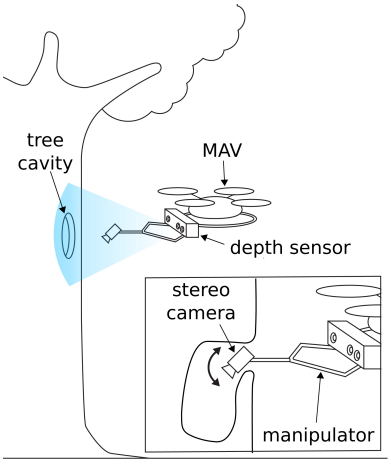


Fig. 1: A MAV equipped with a dexterous manipulator performing a tree cavity inspection using a depth sensor to detect the cavity and a close-up view of the end-effector of the manipulator fitted with a stereo camera being inserted into a cross section of the cavity.

[1]Kelly Steich and Paul Beardsley are with Disney Research Zurich, Stampfenbachstrasse 48, 8006, Zurich, Switzerland. steichk@student.ethz.ch

[2]Kelly Steich, Mina Kamel and Roland Siegwart are with the Autonomous Systems Lab at ETH Zurich, Leonhardstrasse 21, 8092, Zurich, Switzerland. mina.kamel@mavt.ethz.ch

[3]Martin K. Obrist and Thibault Lachat are with the Swiss Federal Institute for Forest, Snow and Landscape Research WSL, Zürcherstrasse 111, 8903, Birmensdorf, Switzerland. thibault.lachat@wsl.ch

*This work has received funding partially from the WSL tree cavity inspection project and the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement No.644128, AEROWORKS.

The *contributions* of this paper are two key enabling technologies for the tree cavity inspection robotic platform, firstly a vision-based system for the detection of cavities of previously unknown size and secondly a strategy for high level control of the MAV and robot manipulator. The system runs semi-autonomously - an initial user input starts the autonomous cavity detection and a pilot manually flies the vehicle to a starting position approximately in front of the cavity, before switching to autonomous control. A more sophisticated system would handle obstacle avoidance of tree branches when approaching a cavity to avoid the need for manual piloting, but this was outside the scope of this work.

The remainder of the paper is organized as follows: Section II describes the system and the tree cavity inspection procedure. The cavity detection algorithm and high level control strategies are detailed in Section III. In Section IV, we discuss a comparison between three candidate sensors to perform the cavity detection and the proposed solution is evaluated in simulation and real–world experiment. Finally, Section V summarizes and concludes the paper.

II. System Description

A. Hardware

The MAV used in this project is the **AscTec Neo hexacopter** [13]. It comes equipped with an inertial measuring unit (IMU) and an on-board computer with an Intel Core i7-5557U dual-core 3.1 GHz processor and 8 GB RAM, running Linux with ROS. Figure 2 shows a Neo hexacopter platform fitted with the additional sensor suite intended for the tree cavity inspection application.

This includes the **Visual-Inertial Sensor (VI-Sensor)** developed by the Autonomous Systems Lab (ASL) at ETH Zurich and Skybotix AG [14]. This sensor, provides stereo images tightly coupled with a high quality IMU, which will be used for the robot's pose estimation.

The **CamBoard pico flexx** from pmdtechnologies [15], a time-of-flight 3D camera, has been installed for cavity detection and is henceforth referred to as *detection sensor*. It is small (68x17x7.25 mm), has a range of 0.1-4 m, a resolution of 224x171 px and a frame rate of up to 45 fps.

A dexterous 3 degrees of freedom parallel **robot manipulator** is attached to the bottom of the body of the vehicle. For this work, the manipulator's pitch is fixed, which restricts its movement to a plane. The exact configuration of the manipulator is shown in Figure 6b.

An **end-effector** was designed for this application [16] and attached rigidly to one link of the manipulator. It includes AWAIBA's **NanEye stereo camera** [17] (2.2x1.0x1.7 mm), an **LED light** and a **laser**, which allows the use of either stereo vision or structured light for 3D reconstruction. It can be rotated inside the cavity around 2 perpendicular axes (yaw and pitch) using servo motors to allow the sensors to capture the complete interior of the cavity.

B. Software

The software architecture is shown in Figure 3 and is comprised of 3 parts: cavity detection, control and GUI.

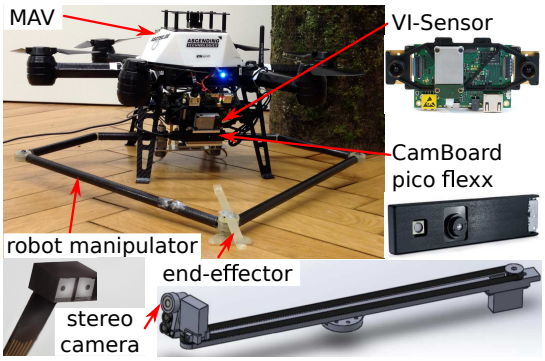


Fig. 2: The hardware comprised of an AscTec Neo hexacopter MAV, a VI-Sensor, a CamBoard pico flexx time-of-flight 3D camera, a parallel robot manipulator and an end-effector fitted with a NanEye stereo camera, an LED light, a laser and servo motors.

All computations of the cavity detection and control run in real time on board the MAV, avoiding communication latency issues in these key parts of the system. The GUI runs on an external processor (e.g., a laptop) to allow the user to interact with the tree cavity inspection process. It communicates with the MAV via WiFi, receiving information of the system's current state and sending back user commands. All software components that are novel contributions of this paper are marked in yellow in Figure 3. They are implemented in C++, using ROS [18], QT [19], OpenCV [20] and PCL [21].

C. Coordinate systems

Figure 3 illustrates the different coordinate frames used by the system. The world frame $\mathcal{W}$ is fixed to the starting position and orientation of the robot, with its z-, y- and x-axes pointing up, left and forward respectively. The vehicle frame $\mathcal{V}$ is rigidly attached to the robot base and has the same axis convention as the world frame. The detection sensor's camera frame $\mathcal{C}$ has a fixed position and orientation relative to the body frame and its origin is at the center of the detection sensor. Its z-, y- and x-axes are pointing forward, right and down respectively. The manipulator frame $\mathcal{M}$ has its origin at the base of the manipulator and its z-, y- and x-axes are pointing up, forward and right respectively.

D. Cavity inspection procedure

The robot is first flown manually, helped by the robot position controller, to a starting point where the cavity is in the field of view of the detection sensor. This can be verified by looking at this sensor's depth image in the GUI.

The user then clicks on this depth image to mark a seed point that belongs to the tree trunk. This starts the cavity detection algorithm, which first analyzes the depth image to measure the 3D position of the cavity in frame $\mathcal{C}$ (see Section III-A). It then analyzes the detection sensor's point cloud to refine this measurement and estimate the cavity's normal. (see Section III-B).

Next, a Kalman filter uses the measured cavity position and the robot pose to generate a continuous estimate of the cavity position in frame $\mathcal{V}$ (see Section III-C).

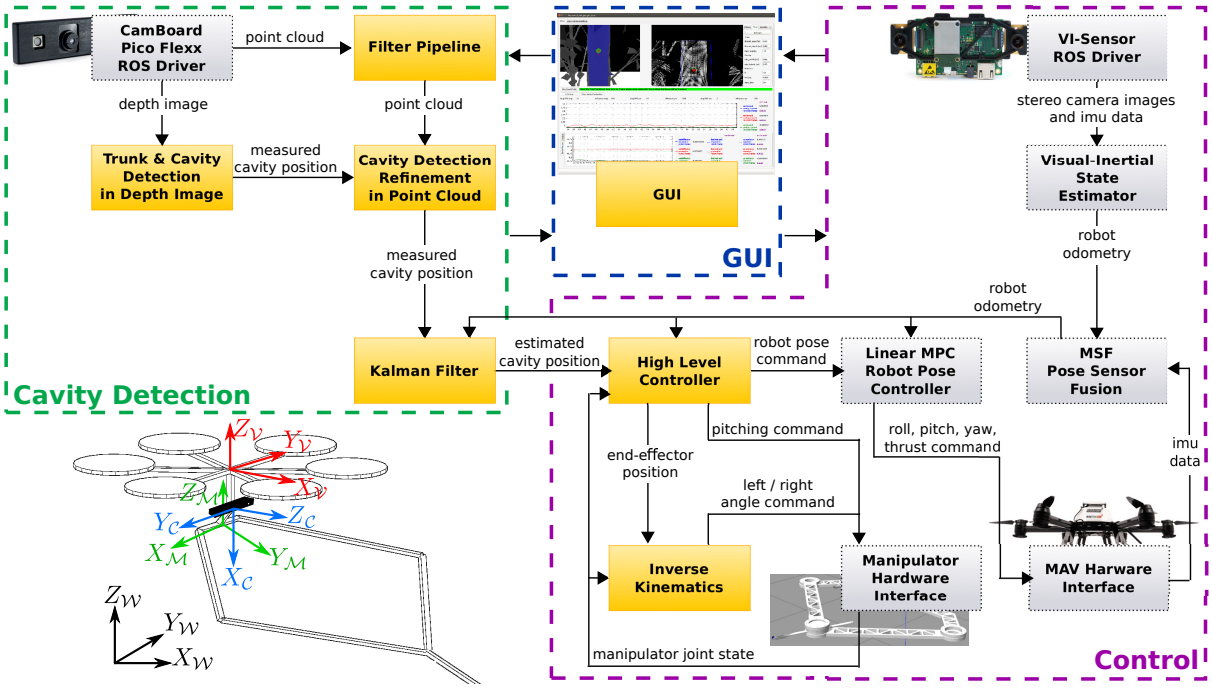


Fig. 3: The software architecture and the defined coordinate frames for the system.

The robot pose in frame $\mathcal{W}$ is estimated using a Robust Visual Inertial Odometry (ROVIO) algorithm, based on the VI-Sensor [22]. This is further fused with the MAV's IMU data using a Multi Sensor Fusion (MSF) approach [23].

Once the GUI shows that the cavity has been successfully detected, the user can activate the auto controller. Subsequently, the high level controller autonomously navigates the MAV to hover directly in front of the cavity entrance at a distance of 0.5 m, by sending commands to the robot position controller. Finally, the user can activate the manipulator controller, which autonomously generates commands to the robot manipulator interface to extend it to insert the end-effector in the cavity (see Section III-D).

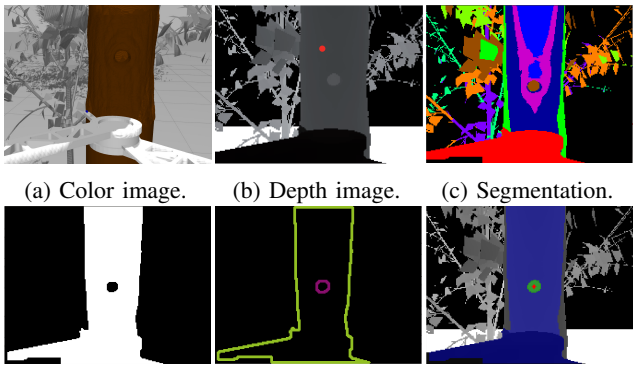
The inverse kinematics derived for this manipulator configuration are used to transform a desired end-effector position command to angle commands for the 2 manipulator servo motors (see Section III-E).

The robot's position controller is a linear Model Predictive Controller (MPC). Using the full state estimate feedback and considering the vehicle dynamics, it provides attitude and thrust reference for the low level controller running on the autopilot provided by the MAV manufacturer. A disturbance observer estimates external forces and moments, such that the linear MPC can compensate for disturbances such as wind.

III. Tree Cavity Inspection

A. Depth image analysis

Figure 4 displays a simulated color image and a depth image from a simulated detection sensor, showing a tree cavity[1] and the different image processing steps performed to detect the tree trunk and the 3D position of the cavity.



(a) Color image. (b) Depth image. (c) Segmentation.

(d) Binarization. (e) Contours. (f) Detection.

Fig. 4: Simulated color image and depth image from a simulated detection sensor, showing a tree cavity and the different image processing steps performed to detect the 3D position of the cavity.

First, the user marks a point in the depth image that belongs to the tree trunk (red point in Figure 4b), to create a seed point $p_s = [x_s, y_s, z_s]^T$, where x_s and y_s are the image coordinates of the marked point and z_s is its depth value.

Segmentation: Then the image is segmented by depth information using K-means clustering with a fixed K (Figure 4c). The cluster with the centroid closest to p_s is assumed to be the cluster representing the tree, C_{tree} (drawn in pink in Figure 4c). Clusters that are closer to C_{tree} than a threshold t_t (drawn in shades of blue in Figure 4c) are merged with C_{tree} to complete the tree cluster. Additionally, we assume that the centroid with the smallest depth always belongs to the cluster representing the robot manipulator C_{man}. Clusters

[1]We use an image from simulation to better illustrate the concepts used, without being distracted by increased noise.

that are closer to C_{man} than a threshold t_m are merged with C_{man} to complete the manipulator cluster (drawn in red in Figure 4c). C_{man} and C_{tree} are merged to form a final tree cluster C_{tm}, which avoids any breaks in the tree cluster due to obstruction by the manipulator cluster.

Binarization: A binary image is created based on whether an image point belongs to C_{tm} and small holes are closed by applying the morphological operations dilation and erosion (Figure 4d).

Contour extraction: Contours are extracted from the binary image using OpenCV's contour detection algorithm [24] (Figure 4e), which also provides the hierarchical relationships of the contours. The contour containing p_s and not surrounded by another contour, is determined to be the tree contour. Any of its children, i.e. all contours contained within the tree contour, are considered potential cavities.

Ellipse fitting: Ellipses are then fitted, in the least-squares sense, to all these children, as woodpecker cavities are always circular or elliptical. This can potentially produce a large number of ellipses, so heuristic filtering is used to select relevant ellipses. They need a minimum width and height (usually between 3-10 cm) and the ratio between width and height cannot be too large. Any ellipses along the very edge of the tree contour are also rejected, since those are most likely to be false positives. Finally, only the ellipse with the largest area E_{max} is selected, since it is least likely to result from noise. Figure 4f shows the tree contour in blue, E_{max} in green and the center of E_{max} in red.

Detection: The center $p_e = [x_{img}, y_{img}]^T$ of E_{max} in image coordinates, together with the average depth of the tree cluster z_{avg} and the intrinsic camera parameters is used to calculate a 3D real world position of the tree cavity $p_c = [x_{real}, y_{real}, z_{real}]^T$ in camera frame $\mathcal{C}$.

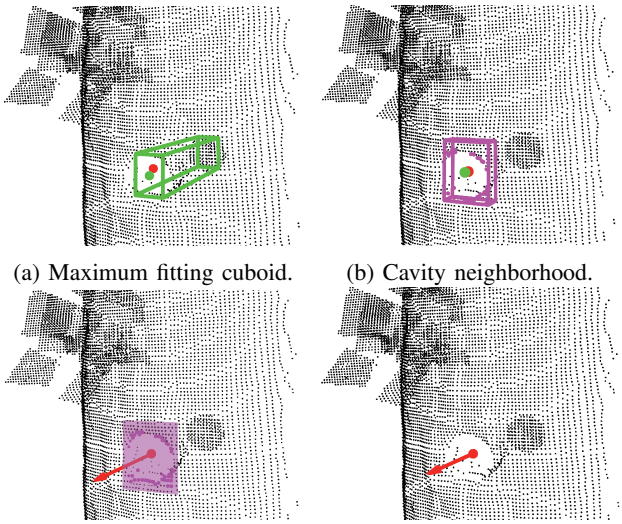
This is repeated for each subsequent depth image frame, with the only difference that a new seed point p_s must be computed at the start. We calculate the moments of the tree contour found in the previous frame to find its center of mass m_c. m_c is selected as a new seed point if its depth is within a threshold t_d to the depth of the previous seed point. Otherwise, the immediate neighborhood of m_c is searched for a point with similar enough depth. The frame is skipped if no new seed point can be found.

B. Cavity detection refinement in the point cloud

Figure 5 displays a point cloud from a simulated detection sensor, showing a tree cavity[2] and the different processing steps for cavity detection refinement.

3D space is partitioned into an octree data structure based on the points in the point cloud generated by the detection sensor. The octree allows us to efficiently determine the points that lie within a certain bounding box. We take advantage of this to both refine the measurement of the 3D position of the tree cavity and determine its normal.

First, we find the maximum fitting cuboid inside the cavity (green cuboid in Figure 5a), to determine with increased certainty the area that is free to insert the end-effector (E_{max} might have been slightly over fitted) and the depth of the cavity. We start with a cuboid centered around the measured position of the tree cavity p_c (red point in Figure 5a) and then iteratively query whether there are any points in an ever increasing cuboid (in positive and negative x and y direction and positive z direction) until a maximum number of inlier points is reached. If the maximum fitting cuboid is smaller than the space necessary for inserting the complete end-effector, this cavity position measurement is rejected, otherwise the center (x_b, y_b) of this maximum fitting cuboid is used to update the measurement with a better estimate $p_c = [x_b, y_b, z_{real}]$ (green point in Figure 5a).



(a) Maximum fitting cuboid. (b) Cavity neighborhood. (c) Fitted plane. (d) Cavity point and normal.

Fig. 5: Point cloud from a simulated detection sensor, showing a tree cavity and the different processing steps performed to refine the detection of the 3D position of the cavity and compute its normal.

Next, the cavity neighborhood points (pink points in Figure 5b) are extracted by querying for all the points lying within a cuboid surrounding p_c (green point in Figure 5b) and larger than the maximum fitting cuboid in x and y direction (pink cuboid in Figure 5b). The z coordinate of the cavity position now equals the average depth of the cavity neighborhood points z_b instead of the average of all the points of the tree. The new cavity position measurement in camera frame $\mathcal{C}$ is $p_c = [x_b, y_b, z_b]$ (red point in figure 5b).

Finally the normal of the cavity N is estimated based on the assumption that the immediate area of the trunk bordering the cavity is approximately planar. A plane is fitted through the cavity neighborhood points (pink plane in Figure 5c) and its normal is assumed to be N (red arrow in Figure 5c).

C. Kalman filter

A Kalman Filter (KF) generates a continuous estimate of the 3D cavity position in frame $\mathcal{V}$. The state $\mathbf{x}_t = [x_t, y_t, z_t]$ of the KF is the 3D cavity position in robot frame at time t and it is initialized as: $\mathbf{x}_0 = [x_0, y_0, z_0] = p_{c0}$ where p_{c0} is the first measurement of the 3D cavity position in frame $\mathcal{C}$

[2]Again, we use a point cloud from simulation to better illustrate the concepts used, without being distracted by increased noise.

transformed to frame $\mathcal{V}$. The underlying model of the KF is given by:

$$\mathbf{x}_t = {}^{\mathcal{V}_t}_{\mathcal{V}_{t-1}}\mathbf{R} \cdot \mathbf{x}_{t-1} + {}^{\mathcal{V}_t}_{\mathcal{V}_{t-1}}\mathbf{T} + \mathbf{w}_{t-1} \tag{1a}$$

$$\mathbf{z}_t = \mathbf{x}_t + \mathbf{v}_t \tag{1b}$$

where $\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$ is the process noise, $\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{J}_t)$ is the measurement noise and ${}^{\mathcal{V}_t}_{\mathcal{V}_{t-1}}\mathbf{R}$ and ${}^{\mathcal{V}_t}_{\mathcal{V}_{t-1}}\mathbf{T}$ are the rotation and translation matrix respectively from robot frame at time $t-1$ to robot frame at time t. They can be expressed in terms of the robot pose at time t and $t-1$ as follows

$${}^{\mathcal{V}_t}_{\mathcal{V}_{t-1}}\mathbf{R} = {}^{\mathcal{W}}_{\mathcal{V}_t}\mathbf{R}^{-1} \cdot {}^{\mathcal{W}}_{\mathcal{V}_{t-1}}\mathbf{R} \tag{2a}$$

$${}^{\mathcal{V}_t}_{\mathcal{V}_{t-1}}\mathbf{T} = {}^{\mathcal{W}}_{\mathcal{V}_t}\mathbf{R}^{-1} \cdot ({}^{\mathcal{W}}_{\mathcal{V}_{t-1}}\mathbf{T} - {}^{\mathcal{W}}_{\mathcal{V}_t}\mathbf{T}) \tag{2b}$$

where ${}^{\mathcal{W}}_{\mathcal{V}_t}\mathbf{R}$ is the rotation from frame $\mathcal{V}$ at time t to frame $\mathcal{W}$, which corresponds to the robot orientation at time t and ${}^{\mathcal{W}}_{\mathcal{V}_t}\mathbf{T}$ is the translation from frame $\mathcal{V}$ at time t to frame $\mathcal{W}$, which corresponds to the robot position at time t.

$\mathbf{z}_t$ is the measurement of the 3D cavity position in frame $\mathcal{V}$ and is given by

$$\mathbf{z}_t = {}^{\mathcal{V}}_{\mathcal{C}}\mathbf{R} \cdot p_{ct} + {}^{\mathcal{V}}_{\mathcal{C}}\mathbf{T} \tag{3a}$$

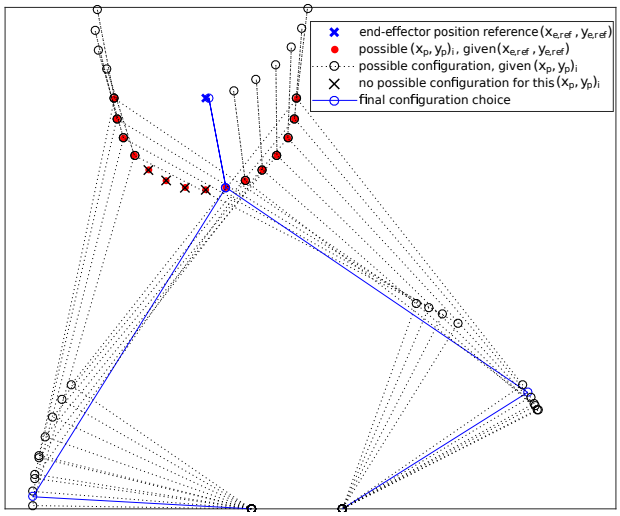
where ${}^{\mathcal{V}}_{\mathcal{C}}\mathbf{R}$ and ${}^{\mathcal{V}}_{\mathcal{C}}\mathbf{T}$ are the fixed rotation and translation from frame $\mathcal{C}$ to frame $\mathcal{V}$ respectively and p_{ct} is the measurement of the 3D cavity position in frame $\mathcal{C}$ at time t.

The prediction step advances the state, i.e. the estimated cavity position in frame $\mathcal{V}$, at each time step (whenever there is new robot pose information, i.e. at 100Hz), while the measurement step incorporates the measurement of the cavity position from the vision-based cavity detection system whenever one is available. This allows the system to estimate the cavity position for several seconds even without new vision measurements and reduces the impact of false cavity detections. The output of the KF is a continuous cavity position estimate in vehicle frame $\mathcal{V}$ at 100Hz.

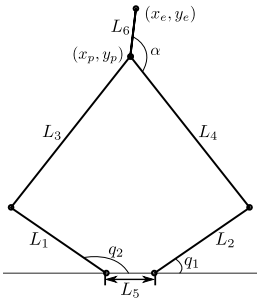
D. Controller

Once the auto controller is started by the user, the high level controller calculates the currently desired position of the robot in frame $\mathcal{W}$ at each time step, using the estimated cavity position and the desired cavity position (0.5,0,0) in frame $\mathcal{V}$ and the current robot pose in frame $\mathcal{W}$. With the cavity normal transformed into frame $\mathcal{W}$ using the current robot pose, the desired robot heading in frame $\mathcal{W}$ is computed. The desired robot position and heading are sent as a command to the linear MPC robot pose controller.

When the user starts the arm controller, the high level controller also sends out commands for the desired end-effector position in frame $\mathcal{V}$, i.e. the estimated cavity position in frame $\mathcal{V}$ with the addition of the depth of the cavity in x direction, filtered trough a first order filter. They are transformed into angle commands using inverse kinematics and sent to the manipulator hardware interface.



(a)



(b)

Fig. 6: The manipulator configuration and a visualization of the inverse kinematics process for a planar delta manipulator with an end-effector.

E. Inverse Kinematics

Figure 6b shows the configuration of the robot manipulator. We define the manipulator joint lengths $L_j, j = 1, ..., 6$, the manipulator angles $q_m = (q_1, q_2)$, the fixed angle between end-effector and one manipulator link α, the manipulator closing position $c = (x_p, y_p)$ and the end-effector position $e = (x_e, y_e)$ The forward and inverse kinematics of a planar delta manipulator have been described in other works [25], [26]. Expanding the forward kinematics to include the end-effector is straightforward. To expand the inverse kinematics we use a "brute force" method. Given a desired end-effector position e_{ref} (blue cross in Figure 6a), we compute the possible manipulator closing positions c_i (red points in Figure 6a) given e_{ref} as the points lying on a semi-circle with center e_{ref} and radius L_6. Increasing the number of semi-circle points i results in higher accuracy in exchange for higher computation time. For all c_i the resulting angles $q_{m,i}$ are calculated (if possible) using the known inverse kinematics for a planar delta manipulator. Using forward kinematics, the resulting end-effector positions e_i are calculated. Finally, the configuration with the minimal error between e_i and e_{ref} is chosen (drawn in blue in Figure 6a) with the corresponding angles $q_{m,i}$.

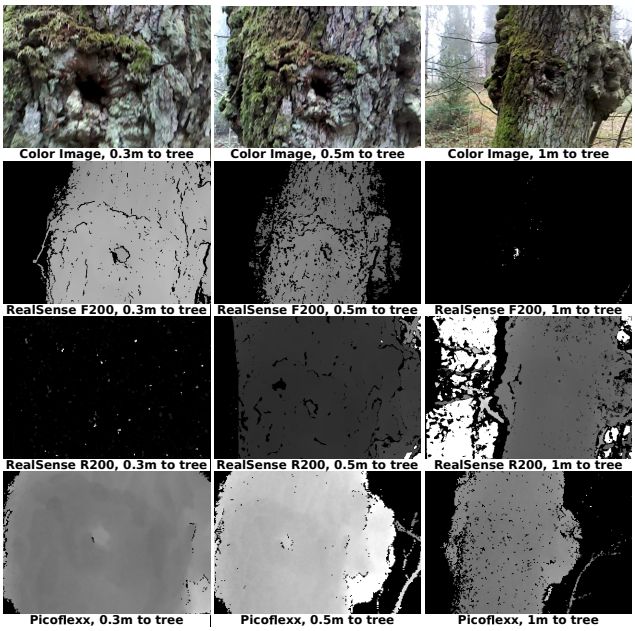


Fig. 7: Comparison of depth images acquired by the Intel RealSense Camera F200 (2nd row), Intel RealSense Camera R200 (3rd row) and pmdtechnologies CamBoard pico flexx (4th row) of the same tree cavity at approximately 0.3m (1st column), 0.50m (2nd column) and 1m (3rd column) while mounted on a telescopic pole on a field trip in the forest . The first row shows a color image of the cavity taken at the corresponding distance from the tree.

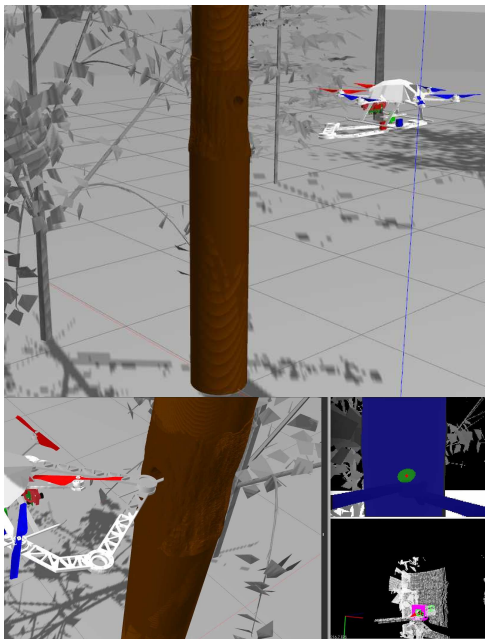


Fig. 8: Simulation experiment setup and result.

TABLE I: MAE and SD for the estimated cavity position compared to the ground truth from the simulation experiment.

	X	Y	Z
MAE [m]	0.0056	0.0011	0.0079
SD [m]	0.0026	0.0014	0.0035

IV. Experiments

A. Sensor comparison

The CamBoard pico flexx was chosen as detection sensor for this application for its small size, large range and good performance outdoors, after comparing it with 2 other depth sensors, the Intel RealSense Camera F200 and the Intel RealSense Camera R200 [27], on a field trip in the forest.

Figure 7 shows a comparison of depth images acquired by all 3 sensors of the same tree cavity at different distances, approximately 0.3m, 0.5m and 1m, while mounted on a telescopic pole. The CamBoard pico flexx works well at all 3 distances, which represent most likely operational range for the tree cavity inspection application. The other sensors fail either at close or far range. Additionally the CamBoard pico flexx is the smallest of the 3 sensors and produced results with very little noise outdoors. Tests were done under an overcast sky, so sunlight was not a factor in the poorer quality results obtained from the two RealSense sensors.

B. Simulation studies

We first test our approach in the Gazebo–based simulation environment RotorS [28]. RotorS is used to simulate a MAV based on the provided model of the AscTec Firefly hexacopter [13] with a VI-Sensor, a CamBoard pico flexx, a robot manipulator with an end-effector and the environment which the simulated MAV operates in. The latter is populated with a realistic 3D model of a tree cavity and tree models in the background, as shown in Figure 8.

In the simulated scenario, the vehicle starts on the ground 1m in front of the cavity and then flies to several manually set waypoints, where the visual of the cavity is sometimes completely lost. Then the auto controller is started and the MAV is navigated to hover 0.5m in front of the cavity. Finally, the arm controller is started and the manipulator is extended to insert the end-effector in the cavity. Figure 8 shows that the end-effector is correctly inserted in the cavity in these experiments.

To evaluate our cavity detection approach, the estimated cavity position is tracked during this whole scenario and compared to the ground truth, as shown in Figure 9. In the simulation experiments, the ground truth robot pose was used as input for the KF instead of the pose estimated by MSF. The red curve represents the KF estimate of the cavity position, while the blue dots represent the measured cavity position from vision-based detection and the black curve shows the ground truth. The filtered position estimate remains sufficiently accurate even if the measured position fluctuates heavily or if there is no result from vision-based detection at all for a certain time window (either because the cavity was not detected correctly or was not in the field of view of the detection sensor anymore).

The mean absolute error (MAE) and the standard deviation of the absolute error (SD) can be found in Table I. The MAE is below 1cm in all directions. The very low error is partly due to using the ground truth robot pose as input

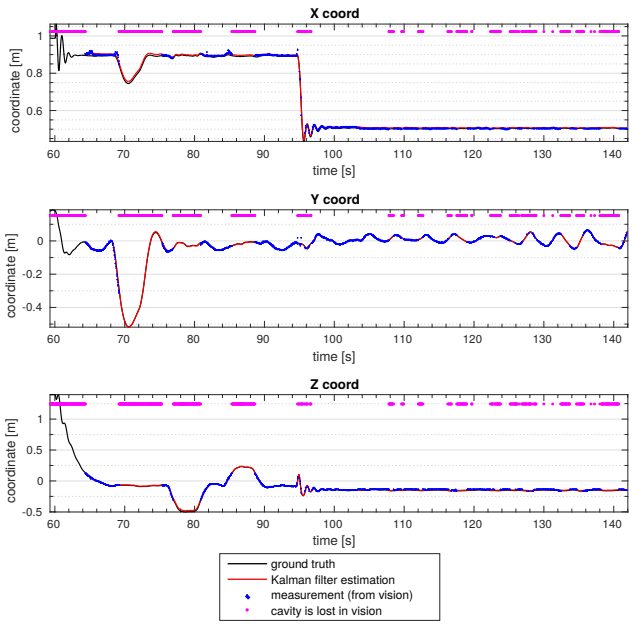


Fig. 9: Comparison of the estimated cavity position, the measured cavity position from the vision-based detection and the ground truth in frame $\mathcal{V}$ during the simulation experiment.

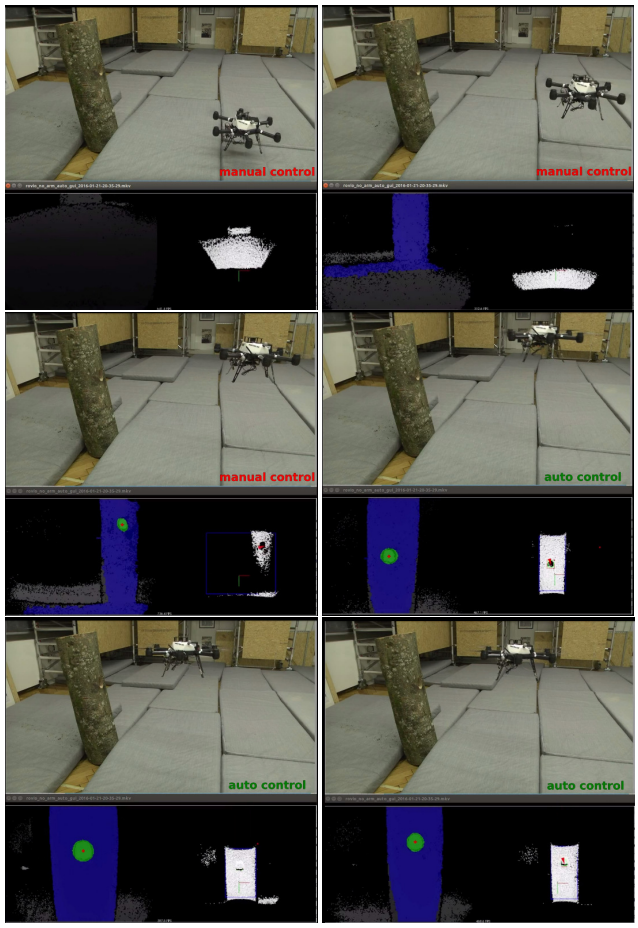


Fig. 10: Sequence from the real-world experiment.

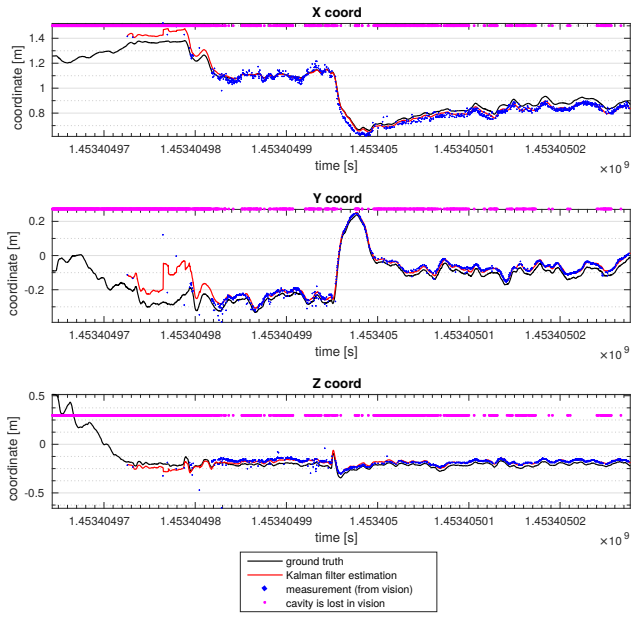


Fig. 11: Comparison of the estimated cavity position, the measured cavity position from the vision-based detection and the ground truth in frame $\mathcal{V}$ during the real-world experiment.

TABLE II: MAE and SD for the estimated cavity position compared to the ground truth from the real-world experiment.

	X	Y	Z
MAE [m]	0.0302	0.0381	0.0321
SD [m]	0.0205	0.0372	0.0088

in the KF. We can still conclude that our proposed approach effectively gives an accurate and continuous estimate of the cavity position and performs satisfactory enough that we can move on to real-world experiments.

C. Experimental evaluation

Real-world experiments, with the aerial robot platform described in II-A, were conducted to further evaluate the performance of the proposed cavity inspection platform. The experiments were conducted in an indoors flying room and we used a model of a tree cavity carved from wood at the Swiss Federal Institute for Forest, Snow and Landscape Research (WSL) The robot manipulator could not yet be tested at the time.

Figure 10 shows a sequence from the experiment, with a view of the GUI showing the cavity detection in the depth image and point cloud. The MAV is first flown up manually and then the user starts the cavity detection by setting a seed point in the depth image to indicate the tree trunk. Once it is clear that the cavity has been successfully detected, the auto controller is activated and the MAV autonomously flies to the cavity and hovers 0.5m in front of it.

The experiments show that the cavity is successfully detected and the MAV is able to hold its position in front of the cavity entrance for several minutes.

Figure 11 shows a comparison of the KF cavity position estimate (red curve) with the measured cavity position from

the vision-based detection (blue dots) and the ground truth (black curve). Although the ground truth is, in this case, only an approximation calculated by transforming the approximately known position of the cavity in world frame $\mathcal{W}$ to frame $\mathcal{V}$, using the estimated robot pose, since no real ground truth information was available. The MAE and SD for the real world experiments can be found in Table II. The MAE is below 4cm in all directions. However, the ground truth is an approximation, therefore the error needs to be regarded critically. Still, we can observe that again the cavity position estimate smoothly follows the ground truth even if there is no result from the vision-based detection at all for a certain time window. Also, the cavity position is quickly corrected again after there were a lot of false positives, e.g., in the very beginning of the experiment, when the MAV is still quite far away from the cavity.

Hence, we can conclude that our approach is able to provide relatively accurate position estimates in real-time.

V. CONCLUSIONS

We have presented an aerial robotic platform for autonomous remote tree cavity inspection, based on a hexacopter MAV equipped with a dexterous manipulator. The full goal of the project is to detect a tree cavity, hover in front of the cavity, insert an end-effector fitted with a stereo camera, and capture data for offline 3D reconstruction of the cavity. This paper has focused on the problem of cavity detection and provided strategies for high level control of the MAV and robot manipulator with minimal user intervention. Simulation experiments show that our approach is able to accurately detect a cavity, navigate the MAV to hover in front of it and insert the end-effector safely. We have carried out real-world experiments to show that accurate cavity detection is possible in real time on board the MAV.

The next stage of the work is to experiment with the manipulator and end-effector using the wooden model of a cavity in an indoors flying room with an external motion capture system (VICON) for better control. This should be followed by outdoor experiments on natural trees.

This paper demonstrates the potential for autonomous aerial robots to support conservation research and offer faster information collection than is possible with manual fieldwork.

REFERENCES

[1] N. Metni and T. Hamel, "A UAV for bridge inspection: Visual servoing control law with orientation limits," *Automation in Construction*, vol. 17, no. 1, pp. 3–10, November 2007.

[2] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2015, (accepted).

[3] G. Darivianakis, K. Alexis, M. Burri, and R. Siegwart, "Hybrid predictive control for aerial robotic physical interaction towards inspection operations," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 53–58.

[4] A. Girard, A. Howell, and J. Hedrick, "Border patrol and surveillance missions using multiple unmanned air vehicles," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, 2004.

[5] P. Rudol and P. Doherty, "Human body detection and geolocalization for uav search and rescue missions using color and thermal imagery," in *Aerospace Conference, 2008 IEEE*, 2008, pp. 1–8.

[6] E. R. Hunt, W. D. Hively, S. J. Fujikawa, D. S. Linden, C. S. T. Daughtry, and G. W. McCarty, "Acquisition of nir-green-blue digital photographs from unmanned aircraft for crop monitoring," *Remote Sensing*, vol. 2, no. 1, pp. 290–305, 2010.

[7] G. Darivianakis, K. Alexis, M. Burri, and R. Siegwart, "Hybrid predictive control for aerial robotic physical interaction towards inspection operations," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 53–58.

[8] M. Fumagalli, R. Naldi, A. Macchelli, F. Forte, A. Keemink, S. Stramigioli, R. Carloni, and L. Marconi, "Developing an Aerial Manipulator Prototype: Physical Interaction with the Environment," *Robotics Automation Magazine, IEEE*, vol. 21, no. 3, pp. 41–50, Sept 2014.

[9] J. Remm and A. Lõhmus, "Tree cavities in forests - the broad distribution pattern of a keystone structure for biodiversity," *Forest Ecology and Management*, vol. 262, no. 4, pp. 579 – 585, 2011.

[10] R. Bütler, T. Lachat, L. Larrieu, and Y. Paillet, "Habitat trees: key elements for forest biodiversity," in *Integrative approaches as an opportunity for the conservation of forest biodiversity*, D. Kraus and F. Krumm, Eds. Freiburg: European Forest Institute, 2013, pp. 84–91.

[11] T. Ranius, "Influence of stand size and quality of tree hollows on saproxylic beetles in sweden," *Biological Conservation*, vol. 103, no. 1, pp. 85 – 91, 2002.

[12] J. Müller, A. Jarzabek-Müller, H. Bussler, and M. M. Gossner, "Hollow beech trees identified as keystone structures for saproxylic beetles by analyses of functional and phylogenetic diversity," *Animal Conservation*, vol. 17, no. 2, pp. 154–162, 2014.

[13] Ascending Technologies GmbH. [Online]. Available: http://www.asctec.de/

[14] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart, "A synchronized visual-inertial sensor system with fpga pre-processing for accurate real-time slam," in *Robotics and Automation (ICRA), IEEE International Conference on*, 2014.

[15] Pmdtechnologies AG. [Online]. Available: http://www.pmdtec.com/

[16] D. Goglio, "Design of end-effector for tree cavity inspection," Semester Thesis, Swiss Federal Institute of Technology Zurich (ETHZ), Switzerland, 2015.

[17] Awaiba. [Online]. Available: http://www.awaiba.com/

[18] Robot Operating System. [Online]. Available: http://www.ros.org

[19] Qt framework. [Online]. Available: http://www.qt.io

[20] OpenCV. [Online]. Available: http://opencv.org

[21] Point Cloud Library. [Online]. Available: http://pointclouds.org

[22] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Sept 2015, pp. 298–304.

[23] S. Lynen, M. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to mav navigation," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, Nov 2013, pp. 3923–3929.

[24] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, pp. 32–46, 1985.

[25] X.-J. Liu, J. Wang, and H.-J. Zheng, "Optimum design of the 5r symmetrical parallel manipulator with a surrounded and good-condition workspace," *Robotics and Autonomous Systems*, vol. 54, no. 3, pp. 221 – 233, 2006.

[26] H. Yu, "Modeling and control of hybrid machine systems — a five-bar mechanism case," *International Journal of Automation and Computing*, vol. 3, no. 3, pp. 235–243.

[27] Intel RealSense. [Online]. Available: http://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html

[28] RotorS: An MAV gazebo simulator. [Online]. Available: https://github.com/ethz-asl/rotors_simulator