Robust visual localization of a UAV over a pipe-rack based on the Lie group SE(3)

Vincenzo Lippiello, *Senior Member, IEEE* and Jonathan Cacace

***Abstract*—Visual inspection and maintenance of industrial pipes using robots represent an emerging application in Oil & Gas and refinery facilities. In this domain, we present a pose tracking system based on a single camera sensor to localize an unmanned aerial vehicle operating over a pipe-rack to carry out inspection activities. We propose a unified framework based on the Lie group SE(3) which allows the simultaneous estimation of the pose of the UAV along with some parameters of the pipe-rack model. Numerical simulations have been performed to demonstrate the effectiveness of the proposed approach.**

SUPPLEMENTARY MATERIAL

Video is available at: https://youtu.be/abeBFLkDW78. Code can be found at: https://github.com/prisma-lab/vs-pipe-rack.

I. Introduction

Pipelines carrying fluids like liquid chemicals, steam, heating water/oil, and similar are generally laid between different units in any petrochemical, power, or chemical processing plants. Typically, they are grouped in a steel-framed structure called *pipe-racks*. To enable easy access for maintenance and preserve ground space, pipe-racks are placed on elevated structures. Pipeline damages can emerge as a weakening of the external covering, as well as in leaks of heat energy, in addition to pipe breaks and losses of the transported medium. At the same time, dangerous situations (like explosions or chemical incidents) can be caused by wear on the thickness of piping through internal corrosion during the transport of aggressive liquids or mechanical stressing in the case of the transport of non-homogeneous materials with solid particles. To avoid risky situations arising from piping defects, pipelines must be regularly inspected through complex and risky operations. Considering that pipe-racks often extend for miles and are located at several meters in height, visually checking all the sections of the pipes is a highly demanding task. Alternatively, built-in measuring systems like pressure or flow meters should be used but, such tools are too expensive and unreliable because even they cannot cover every high-risk area. In this domain, thermographic pipeline inspection represents an advanced technique to detect pipe damages. Since the temperature of the material transported by pipe differs from that of the surrounding environment, thermal cameras can play an important role in the inspection and maintenance tasks. The use of autonomous robots, like Unmanned Aerial Vehicles (UAVs), equipped with vision and thermal sensors represent a reliable and low-cost solution for inspection tasks ([1] [2]). An aerial vehicle can fly above the whole surface of a pipe rack while transferring relevant information to ground operators. To automatize such an inspection task, different challenges must be tackled. First, the drone should be able to stabilize during the flight autonomously. In this context, GPS sensors could be unreliable due to the vicinity of tall iron structures. Then, measured data should be localized with respect to a local reference frame with sufficient accuracy and repeatability to perform continuous monitoring and trending.

Authors were with the Department of Electrical Engineering and Information Technology, University of Naples Federico II, Naples, 80125 ITALY e-mail: {vincenzo.lippiello, jonathan.cacace}@unina.it.

This work was supported by the HYFLIERS and AERIAL-CORE Projects (Horizon 2020) under Grants 779411 and 871479, respectively.

To face these problems, in this paper, we propose a new robust visual tracking algorithm, which allows a UAV to automatically stabilize its position and orientation over a pipe rack and simultaneously localize its motion with respect to a local reference frame fixed with a pipe rack. In the proposed approach, we assume that the UAV is equipped with a standard camera used to track the motion of the vehicle with respect to a local arbitrary reference frame. This frame will be used to assign the desired inspection paths on a pipe rack. Initially, we assume that the geometry of the pipe-rack is precisely known and the diameter of the pipes. This assumption is reasonable since the shape of the pipelines is standard regulated. However, we extend the proposed approach to the case of structures with uncertain knowledge parameters.

blue In our setup, the geometric model of the scene is exploited to perform the loop closure between the motion of the UAV and its new estimated pose. In particular, the data get by the camera sensor of the UAV are matched with the scene model. The motion is calculated estimating the distance between the detected scene and the projection of the modeled one.

blue The main contribution of this work regards the application of Lie groups and their algebras to a localization problem in the specific domain of inspection of pipe racks in oil and gas facilities. This approach differs from the classical SLAM solutions since the environment is assumed to be already known. At the same time, it differs from sample-based localization techniques like the well-known Monte Carlo Localization since, thanks to the integration of the Lie algebra, image data are exploited to reconstruct the relative pose of the camera, i.e. the drone, with respect to the observed object, i.e. the pipe-rack in our case, based on the provided scene model. Our work takes inspiration from the similar works in which the lie groups and their algebra has been used to perform visual servoing tasks, [3] and [4], in which a position based visual servoing technique has been proposed and demonstrated to achieve high accuracy.

Finally, to assess the effectiveness of the proposed approach, experiments have been in a MATLAB simulation environment consisting of a pipe-rack and a dynamic modeled UAV equipped with a downward camera. During these tests, different operative conditions have been considered.

II. Motivation and Related works

UAVs are versatile systems that can be used in several inspection applications like water flow and crop monitoring, farm animal detection and counting and many more. All these applications require a precise localization of the UAV during the flight that typically is achieved by exploiting GPS sensors. However, this level of localization may not be present in GPS-denied or GPS-spoofed environments where the GPS signal is not reliable (like the case considered in this paper, due to the vicinity of iron structures). A solution could be to use costly GPS devices, like the differential GPS, which improves UAV localization using a second GPS ground station. However, this approach requires modifying the environment and the use of expensive infrastructure that is not always possible. In this context, visual odometry with specific hardware or SLAM techniques can be adopted [5] [6]. On the other hand, these methods need additional hardware like depth sensors or event-based cameras. blueAdditional equipment on the UAV could decrease its overall operating autonomy due to the increase of its payload and the computational effort needed to stream and elaborate sensor data. For these reasons, it is important to extract only small amounts of meaningful information in real-time video streams at the camera frame rate [7]. The use of dense two-dimensional image processing involves a high computational cost, while feature-based tracing is limited to finding strong image features such as *contours* [8]. Contours represent useful features to track for visual servoing applications for several reasons: i) contours can be tracked reliably against cluttered environments [9], ii) the large number of measurements that can be made by integrating normal velocities around the contour allows for accurate measurement of image motion [10], iii) a planar contour can be easily constrained to undergo affine or projective deformations [11] [3].

Several successful systems have been based on tracking the image contours of a known model. The problem of determining the camera motion from apparent contours or silhouettes of a priori unknown curved 3D surfaces has been considered in [12]. A sequence of images shows how to use the generalized epipolar constraints on apparent contours. An accurate algorithm for computing the motion is presented based on a maximum likelihood estimate. A framework for three-dimensional model-based tracking has been presented in [4] [13], in which a graphic rendering technology has been combined with constrained active contour tracing to create a robust wire-frame tracing system. A Lie group formalism has been used to transform the motion computation problem into a simple optimization problem. A visual servoing system has been built and tested using this framework. A vision-based line following strategy for UAVs has been proposed in [14] follow the margins of water channels, crop lines and other similar models.In this context, a nonlinear path following controller has been designed along with a visual-based line detection algorithm capable of detecting the average position and orientation of the main lines on the image frames captured by the UAV downwards facing camera. Closer to our domain, in [15] a transmission line inspection system using a quadrotor UAV has been proposed to improve inspection efficiency. The hardware and software for real-time data processing for transmission line detection and tracking have been presented. In [16], arbitrary object contours in a 2D image have been considered as plane shapes to identify their structures to inspect ground pipelines autonomously with a UAV. Image processing is applied to estimate the object pose, while the UAV autonomously tracks the structure according to this position estimation. A robust autonomous UAV navigation approach along one side of the overhead transmission lines for inspection was presented in [17]. A perspective model was adopted, and a Pan/Tilt monocular-based navigation scheme has been developed. The transmission lines are detected and their vanishing point calculated and optimized to provide the UAV with a robust and accurate heading. A 3D tracking algorithm for oil & gas pipelines based on the tracking structure identified via depth images has been presented in [18]. Through linear fitting of the nearest pixels of the depth image, the pipeline berm can be identified. Then according to the orientation and lateral difference information of the pipeline obtained, the UAV autonomous navigation and tracking can be realized.

III. Method

To compute the camera pose during the autonomous pipeline inspection task, the specificities of the pipe-rack geometry are addressed, characterizing it with a combination of contour image and point image features. An image processing algorithm has been deployed to identify tube contours combined with the well-known Hough transform. Once extracted the transformation of the image features between different consecutive images, the camera pose is calculated linearizing with respect to the image motion. This process is computed in terms of the *Lie group* SE(3) and its *Lie algebra*. In this context, the group SE(3) represents the space of the poses that exactly form the desired output of the algorithm, while Lie algebra is the space tangent to the group to the identity and is, therefore, the natural space in which to represent differential quantities such as velocity and small motion. Therefore, such representation provides a canonical method for linearizing the relationship between pose parameters and image movement. In the following, the models used to characterize the pipe-rack and the UAV camera are discussed.

Without loss of generality, pipe-rack is described as an ordered sequence of n_p pipes with $n_p > 1$ where the pipes are placed on a common plane, numbered from left to right looking in the direction of the flow. Note that, even if the pipelines are not perfectly parallel, it can be assumed that they are local, with respect to the portion of the plane visible in the camera plane. The support plane of the pipelines corresponds to the (x_p, y_p) plane, with the y_p-axis aligned with the flow

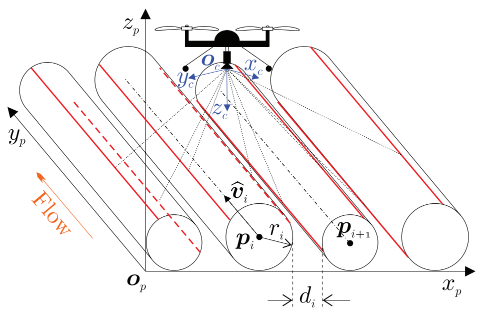

Fig. 1. Representation of the pipe rack and reference frames.

direction and the origin $\boldsymbol{o}_p$ corresponding to the most left point of the first pipe, as shown in Fig. 1. Notice that the origin of the fixed reference frame $\mathcal{P} : \boldsymbol{o}_p - x_p y_p z_p$ can be chosen arbitrarily, e.g. with respect to a local physical reference. Still, in this paper, we assume the drone laying in the (x_p, z_p) plane at time $t = 0$. We denoted with $\boldsymbol{p}_i$ the central point of the i-th pipe, r_i the (external) pipe radius, $\hat{\boldsymbol{v}}_i$ the unit vector aligned with the pipe/flow direction, and d_i di distance between the i-th and $(i+1)$-th pipe.

As for the *camera model*, we assume that the UAV is equipped with a downward-looking camera. A camera reference frame $\mathcal{C} : \boldsymbol{o}_c - x_c y_c z_c$ is introduced, with the z_c-axis corresponding with the *camera optical axis*. To make the notation simpler, let's assume that the centre of the camera reference frame $\boldsymbol{o}_c$ corresponds to the centre of mass of the UAV, thus avoiding the introduction of another reference frame only for the UAV. In this context, the *Pinhole* camera model is adopted. The non-singular upper-triangular *camera matrix*, which contains the internal camera parameters, is denoted as follows:

$$\boldsymbol{K} = \begin{pmatrix} f_c & s_c & u_0 \\ 0 & r_c f_c & v_0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{1}$$

where f_c is the camera focal length in pixels, r_c is the aspect ratio, s_c is the skew, and (u_0, v_0) is the principal point in pixels. The *camera projection matrix* $\boldsymbol{P}$ in the fixed reference frame is represented as the product of the camera matrix and the Euclidean projection matrix $\boldsymbol{E}_p^c$ representing the position and orientation of $\mathcal{P}$ with respect to $\mathcal{C}$ as $\boldsymbol{E}_p^c = \begin{pmatrix} \boldsymbol{R}_p^c & \boldsymbol{p}_p^c \end{pmatrix}$, where $\boldsymbol{R}_p^c \in \mathrm{SO}(3)$ and $\boldsymbol{p}_p^c \in \mathbb{R}^3$ are the rotation matrix and the position of the pipe-rack represented in $\mathcal{C}$, respectively.

Given a Cartesian point $\boldsymbol{p}^p$ fixed in $\mathcal{P}$, the corresponding projective coordinates (*point image feature*) are given by

$$\boldsymbol{r} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \boldsymbol{P}\bar{\boldsymbol{p}}^p, \tag{2}$$

with $\bar{\boldsymbol{q}} = \begin{pmatrix} \boldsymbol{q}^\top & 1 \end{pmatrix}^\top$ the homogeneous representation of a vector, while the actual image coordinates are given by

$$\boldsymbol{m} = \begin{pmatrix} X \\ Y \end{pmatrix} = \frac{1}{w} \begin{pmatrix} u \\ v \end{pmatrix}. \tag{3}$$

The pose of the pipe-rack with respect to the UAV camera can be expressed by the homogeneous transformation between the reference frames $\mathcal{P}$ and $\mathcal{C}$ as follows:

$$\boldsymbol{H}_p^c = \begin{pmatrix} \multicolumn{2}{c}{\boldsymbol{E}_p^c} \\ \mathbf{0_{1\times 3}} & 1 \end{pmatrix}, \tag{4}$$

where $\mathbf{0_{n\times m}}$ is a null $(n \times m)$ matrix. Let the Euclidean transformation matrix $\boldsymbol{M}(t) \in \mathrm{SE}(3)$ be the relative motion of $\mathcal{C}$ between two consecutive camera frames $(t-1) \to t$ with respect to $\mathcal{P}$ and expressed in $\mathcal{P}$. From now on, we will call $\boldsymbol{M}$ the *motion matrix*. Hence, the rigid motion of the UAV camera between consecutive video frames can then be represented as follows:

$$\boldsymbol{H}_p^c(t) = \boldsymbol{H}_p^c(t-1)\boldsymbol{M}(t). \tag{5}$$

One objective of this paper is to elaborate the relationship between $\boldsymbol{M}$ and the apparent motion of the image features in the image plane of the camera (*motion tracking* problem) as described in the following.

Motion Tracking: The motion matrix $\boldsymbol{M}$ forms a 4×4 matrix representation of the SE(3) group of rigid body movements in three-dimensional space, which is a six-dimensional *Lie group*. In this context, the motion is represented considering a coordinate system to describe small transformations close to the identity, where the axes correspond to the different deformation modes and the affine transformations are specified as a weighted sum of the group generators added to the identity. This leads to a local vector space representation for infinitesimal transformations, in which the affine transformation matrix $\boldsymbol{M}$ can be directly obtained from a vector (see [19] [20]). Possible *generators* of the SE(3) group are the elementary translations/rotation in/around the x, y and z axes:

$$\boldsymbol{G}_1 = \begin{pmatrix} 0&0&0&1\\0&0&0&0\\0&0&0&0\\0&0&0&0 \end{pmatrix}, \quad \boldsymbol{G}_2 = \begin{pmatrix} 0&0&0&0\\0&0&0&1\\0&0&0&0\\0&0&0&0 \end{pmatrix}, \quad \boldsymbol{G}_3 = \begin{pmatrix} 0&0&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&0 \end{pmatrix},$$
$$\boldsymbol{G}_4 = \begin{pmatrix} 0&0&0&0\\0&0&-1&0\\0&1&0&0\\0&0&0&0 \end{pmatrix}, \boldsymbol{G}_5 = \begin{pmatrix} 0&0&1&0\\0&0&0&0\\-1&0&0&0\\0&0&0&0 \end{pmatrix}, \boldsymbol{G}_6 = \begin{pmatrix} 0&-1&0&0\\1&0&0&0\\0&0&0&0\\0&0&0&0 \end{pmatrix}. \tag{6}$$

Such generators compose a basis for the vector space (known as *Lie algebra*) of the derivatives of SE(3) near the identity. Let μ_i be the six coefficients of the projected vector in the direction of the elementary translations, thus the elements of the group can be obtained from the generators via the *exponential map*:

$$\boldsymbol{M} = \exp(\mu_i \boldsymbol{G}_i) \triangleq \sum_{k=0}^{\infty} \frac{\mu_i^k}{k!} \boldsymbol{G}_i^k, \tag{7}$$

where the Einstein summation convention over Latin indices has been adopted and will be used throughout this paper. Since the motion matrix $\boldsymbol{M}$ is the structure transformation between two adjacent video frames, then the tracking problem becomes to find the μ_i coefficients that describe the inter-frame transformation. If the camera frame rate is sufficiently high with respect to the apparent motion of the image features in the image plane, i.e. if $\mu_i \ll 1$, then the exponential map can be approximated by its linear terms as $\boldsymbol{M} \approx \boldsymbol{I}_4 + \mu_i \boldsymbol{G}_i$, where the symbol $\boldsymbol{I}_n$ represents the $(n \times n)$ identity matrix. If this is the case, the motion is approximated as a linear sum of that produced by each of the generators.

The partial derivative of the projective coordinates of the image features, corresponding to a Cartesian point $\boldsymbol{p}^p$ fixed

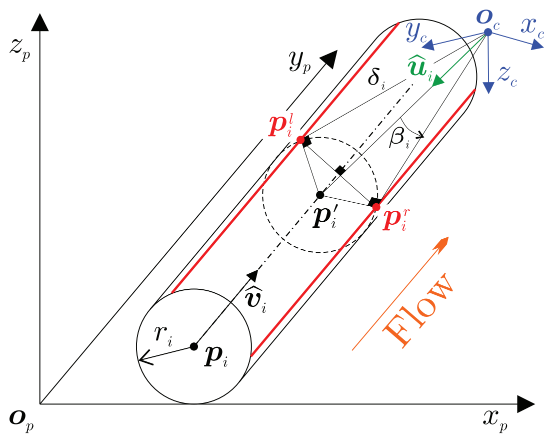

Fig. 2. Visible edges of a pipeline from the UAV camera.

with respect to $\mathcal{P}$, with respect to the *i-th* generator motion can be calculated as

$$\boldsymbol{r}_i' = \begin{pmatrix} u_i' \\ v_i' \\ w_i' \end{pmatrix} = \boldsymbol{P}\boldsymbol{G}_i\bar{\boldsymbol{p}}^p, \tag{8}$$

that are linear functions of position, while the corresponding motion in true image coordinates is given by

$$\boldsymbol{\xi}_i = \begin{pmatrix} X_i' \\ Y_i' \end{pmatrix} = \frac{1}{w_i}\begin{pmatrix} u_i' - \frac{u_i w_i'}{w_i} \\ v_i' - \frac{v_i w_i'}{w_i} \end{pmatrix}, \tag{9}$$

which corresponds to the vector fields in the image plane used to compute the affine transformation that describes the structure transformation in the camera's view.

The pipe-rack is characterized by a combination of two kinds of image features: *Pipeline edges* and *local texture features*.

Pipeline edges: Edges represent the most reliable image features that can be extracted from a pipeline. They correspond to a couple of straight lines depending on the relative pose of the camera (the viewpoint) relative to the pipe. Assuming to perform the estimation of the pose of the UAV camera using the proposed motion tracking algorithm, visible edges of each pipe belonging to the pipe-rack can be projected onto the camera image plane (model rendering) to extract an error vector describing the structure deformation. Let $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_i^r$ be points located on the visible lines, specifically on the left line and on the right line (looking towards the flow direction), respectively, at a minimum distance from the camera (see Fig. 2). From now on, unless expressly indicated, all quantities refer to $\mathcal{P}$. We consider the point $\boldsymbol{p}_i'$ laying on the pipe axis and at the minimum distance with respect to the camera, thus

$$\boldsymbol{p}_i' = \boldsymbol{p}_i + (\boldsymbol{o}_c - \boldsymbol{p}_i)^\top \hat{\boldsymbol{v}}_i, \tag{10}$$

and the unit vector pointing from the camera to such a point

$$\hat{\boldsymbol{u}}_i = \frac{1}{l_i}(\boldsymbol{p}_i' - \boldsymbol{o}_c), \tag{11}$$

where $l_i = ||\boldsymbol{p}_i' - \boldsymbol{o}_c||$ corresponds to the minimum distance between the camera and the pipe principal axis. Then, by

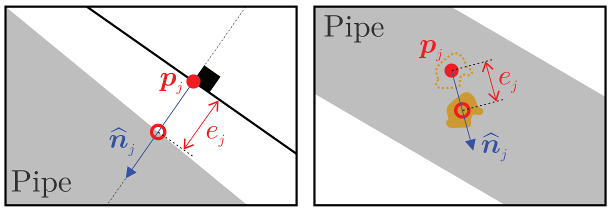

Fig. 3. Extraction of the normal component of the apparent motion of pipe edge (left) and of a pipe texture point feature (right).

computing the angle $\beta_i = \arcsin(r_i/l_i)$ and the distance from the camera to the visible lines

$$\delta_i = ||\boldsymbol{o}_c - \boldsymbol{p}_i^l|| = ||\boldsymbol{o}_c - \boldsymbol{p}_i^r|| = \sqrt{l_i^2 - r_i^2}, \tag{12}$$

we can finally compute the points on the visible edges as

$$\begin{aligned} \boldsymbol{p}_i^l &= \boldsymbol{o}_c + \delta_i \boldsymbol{R}(\hat{\boldsymbol{v}}_i, \beta_i)\hat{\boldsymbol{u}}_i & (13)\\ \boldsymbol{p}_i^r &= \boldsymbol{o}_c + \delta_i \boldsymbol{R}(\hat{\boldsymbol{v}}_i, -\beta_i)\hat{\boldsymbol{u}}_i, & (14) \end{aligned}$$

where $\boldsymbol{R}(\hat{\boldsymbol{a}}, \alpha)$ is the rotation matrix around the axis $\hat{\boldsymbol{a}}$ of an angle α [21]. Hence, both visible lines are fully described in Cartesian space through these points and the unit vector $\hat{\boldsymbol{v}}_i$. Finally, the back projection on the camera image plane is a straightforward operation that only requires care in considering the camera's field of view by calculating the two ends of the visible segment of these lines.

Fig. 3 (left) shows the process for extracting the normal component of the apparent motion of the tube edges. In particular, the rendered segments are sampled with some measurement points and, starting from each of these points $\boldsymbol{p}_j$, the closest pipe edge is searched along the direction normal to the rendered segment, which corresponds to the unit vector $\hat{\boldsymbol{n}}_j$. In this way, the extraction process has only linear complexity, instead of a square complexity as with other types of image features. Finally, the distance e_j between the measurement point and the actual pipe edge is measured.

Texture features: Due to the symmetry of a pipe-rack along the flow direction, using only the edges, the movement along the y_p axis is not observable. To address this problem, the second type of image feature has to be considered. A real pipeline is typically covered in rust and dirt, which can be easily identified and tracked with standard image descriptors based on SURF, SIFT, BRIEF or ORB [22]. At the start of the tracking process, all the point features that are visible on each pipe, which are delimited in the image by the visible edges computed in the previous section, are identified. Through the intersection of the corresponding camera ray and the pipe 3D model, it is possible to estimate the 3D position of such model with respect to $\mathcal{P}$. These points are added to the 3D model of the pipe rack along with the corresponding image descriptors, which are needed to locate the same points again in future images. Once a new image needs to be processed, two processes begin: the first is dedicated to finding previous feature points around the expected position of the rendered model. Differently, the second process is dedicated to finding new image features to add to the 3D model to cope with the continuous motion of the UAV along the pipe-rack.

For all matched points $\boldsymbol{p}_j$, as shown in Fig. 3(right), the apparent motion is described through its direction by means of

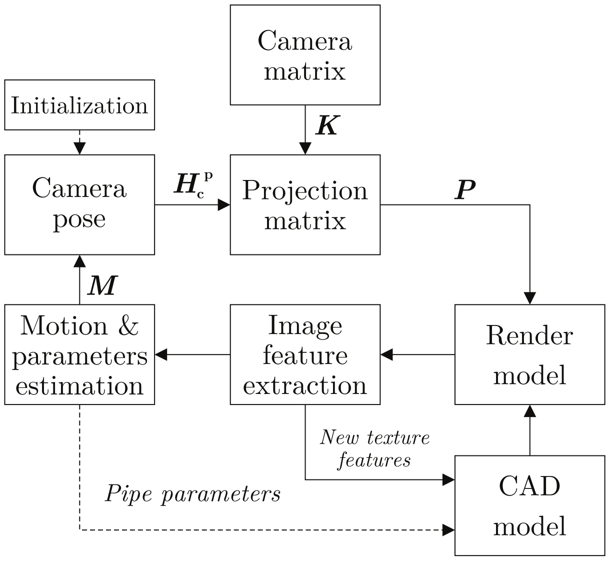

Fig. 4. Block scheme of the motion tracking (solid lines) and of the model parameters adaptation processes (dotted lines).

the unity vector $\hat{\boldsymbol{n}}_j$, and the corresponding image displacement e_j.

The overall tracking process is represented in the block scheme reported in Fig. 4. At each sampling time, the expected view of the pipe-rack is rendered ("CAD model"), and the location of each visible point image feature is identified using the current estimation of the camera projection matrix $\boldsymbol{P}(t-1)$ ("Render Model"). A predefined number of sampling points with a regular interval are located on such visible edges, while suitable elaboration windows are extracted from the image corresponding to each point image feature. The actual edge is then searched in the image for each sampling point for a nearby edge in the normal direction to the rendered edge. Similarly, each point image feature descriptors are searched in the corresponding window, and new key points are identified and assigned to the CAD model of the corresponding pipeline ("Image feature extraction"). In this context, 500 sampling points and tens of point image features are measured in this way, for a total of η image features. The algorithm then projects such η-dimensional measurement vector onto the 6-dimensional subspace corresponding to homogeneous transformation giving the least-squares estimate of the motion matrix $\boldsymbol{M}(t)$ ("Motion & parameters estimation"). The camera UAV pose $\boldsymbol{H}_p^c(t)$ is then update These steps are cyclically repeated to track the pose of the camera in the defined fixed frame.

To calculate the motion of the camera, for each sample point $\boldsymbol{p}_j$, indifferently if it has been extracted from an edge or a texture feature, the normal component of the motion fields are computed as

$$\sigma_{i,j} = \boldsymbol{\xi}_i^\top \hat{\boldsymbol{n}}_j, \tag{15}$$

thus e_j can be fitted as a linear combination of the $\sigma_{i,j}$ to give a linearized estimation of the camera motion. Note that $\sigma_{i,j}$ describes the subspace of the Euclidean transformation group through the magnitude of the edge normal motion and of the point image features that would be observed in the image at the j-th sample point for the i-th group generator. These measurements can be considered as a set of six η-dimensional vectors describing the motion in the image of each of the six Euclidean transformation modes. This motion corresponds to the pipe rack's geometric transformation, which best fits the observed edge and texture-features positions and can be found by minimizing the square error between the transformed and actual edge/texture-feature positions in the image plane. To such a purpose, the least-squares algorithm can be applied as follows:

$$\gamma_i = e_k \sigma_{i,k} \tag{16}$$

$$\boldsymbol{J}_{i,j} = \left(\sigma_{i,k}\sigma_{j,k}\right) \tag{17}$$

$$\mu_i = \boldsymbol{J}_{i,j}^{-1}\gamma_j \tag{18}$$

Proof. Let λ_i be the six coefficients of the projected vector. Let's demonstrate that $\lambda_i = \mu_i$ gives the minimum least-squares solution to

$$\rho = \sum_k \left(e_k - \lambda_j \sigma_{j,k}\right)^2. \tag{19}$$

Let's derive the gradient of (19) with respect to the projection vector

$$\frac{\partial \rho}{\partial \lambda_i} = -2\sum_k \sigma_{i,k}\left(e_k - \lambda_j \sigma_{j,k}\right), \tag{20}$$

and setting $\lambda_i = \mu_i + \epsilon_i$ while substituting (18) gives

$$\begin{aligned}\frac{\partial \rho}{\partial \lambda_i} &= -2\sum_k \sigma_{i,k}\left(e_k - \sigma_{j,k}\left(\boldsymbol{J}_{j,s}^{-1}\sum_r \sigma_{s,r}e_r + \epsilon_i\right)\right)\\ &= -2\sum_k \sigma_{i,k}e_k + 2\boldsymbol{J}_{i,j}\boldsymbol{J}_{j,s}^{-1}\sum_r \sigma_{s,r}e_r + 2\boldsymbol{J}_{i,j}\epsilon_j\\ &= 2\boldsymbol{J}_{i,j}\epsilon_j.\end{aligned} \tag{21}$$

The μ_i are thus the coefficients of a linear approximation to the Euclidean motion which minimizes the sum squared error between the rendered and the observed image features ($\epsilon_i = 0$). When $\epsilon_i \neq 0$, the residual sum-squared error is computed integrating (20)

$$\rho = \bar{\rho}|_{\epsilon_i=0} + \epsilon_i \boldsymbol{J}_{i,j}\epsilon_j. \tag{22}$$

□

The proposed tracking algorithm is therefore able to continually estimate H_p^c (through (5) and, hence, $\boldsymbol{P}$) by computing the coefficients μ_i of inter-frame motions as shown in Fig. 4.

Estimation of pipe-rack Parameters: One key advantage of the proposed approach is that it is possible to include more complex situations such as the estimation of 3D-model parameters along with the object motion. Considering a real pipe-rack is reasonable to assume that the radius of each pipe is known and that the pipes are parallel. However, the pipes are not always constrained, but often resting only on the supports of the pipe holder. Due to thermal expansion and vibrations generated during the liquid transportation operations, the distance between the pipes could vary over time. So, the proposed approach can be extended to estimate online the distance between adjacent pipes. Compared to the reference frame shown in Fig. 1, which is fixed with the first pipe, if the distance between the i-th and the $(i+1)$-th pipe varies of an assigned amount, assuming that the other distances between the pipes remain fixed, all the pipes from the $(i+1)$-th move rigidly along the x_p axis by this amount. So, the visible edges also move similarly. Thus, the projected point onto the camera plane of the j-th feature point belonging to the i-th pipe, with $i = 2, \dots, n_p$, can be expressed as

$$\boldsymbol{r}_{i,j} = \boldsymbol{P}\boldsymbol{T}_i \bar{\boldsymbol{p}}^p_{i,j} \tag{23}$$

where $\bar{\boldsymbol{p}}^p_{i,j}$ represents the nominal position when all pipes are tangent (i.e. $d_i = 0\ \forall i$), represented in $\mathcal{P}$, and

$$\boldsymbol{T}_i = \begin{pmatrix} 1 & 0 & 0 & \bar{d}_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \text{ with } \bar{d}_k = d_1 + \dots + d_k. \tag{24}$$

The generators of $\boldsymbol{T}_i$ are equal and correspond to $\boldsymbol{G}_1$. Let μ_i, with $i = 7, \dots, n_p + 5$, be the coefficients of the projected vector in the direction of the elementary translations which corresponds to $\bar{d}_k$, with $k = 1, \dots, n_p - 1$. The elements of the corresponding Lie group can be obtained from the generator via the exponential map, which can be approximated with the linear term in case of slow variations between consecutive frames:

$$\begin{aligned} \boldsymbol{T}_i(t) &= \exp\left(\mu_{i+5}\boldsymbol{G}_1\right)\boldsymbol{T}_i(t-1) \\ &\approx \left(\boldsymbol{I}_4 + \mu_{i+5}\boldsymbol{G}_1\right)\boldsymbol{T}_i(t-1). \end{aligned} \tag{25}$$

Since the pipe distance is unknown but assumed to remain constant, the optimization problem becomes to find the μ_i coefficients that nullify the inter-frame transformation due to the pipe displacement error. Finally, thanks to the simple structure of $\boldsymbol{T}_i$, equation (25) can be further simplified as follows: $\bar{d}_i(t) = \bar{d}_i(t-1) + \mu_{i+6}$.

The partial derivative of (23) with respect to the $(i+5)$-th generator can be calculated as

$$\boldsymbol{r}'_{i,j} = \boldsymbol{P}\boldsymbol{G}_1\boldsymbol{T}_i \bar{\boldsymbol{p}}^p_{i,j}, \tag{26}$$

with $i = 2, \dots, n_p$.

The corresponding motion in true image coordinates is given by (9), which is the corresponding vector fields in the image plane. These are added to the vector fields already used for the motion tracking algorithm (see Fig. 4), which now fits into a $(n_p + 5)$-dimensional rather than the six-dimensional least-squares solution. In this way, the 3D-model parameters of the pipe-rack can be estimated during the motion of the camera.

Finally, the objective sum-of-squares function previously introduced can be significantly affected by some large-error measurements. Likewise, the corresponding Gaussian distribution dies too quickly to admit many sample measurements with a large number of standard deviations.

A well-known solution is to replace the objective function with one that applies less weighting to the outliers [23] and can be obtained by replacing (16) and (17) with

$$\gamma_i = \sum_k s(e_k) e_k \sigma_{i,k} \tag{27}$$

$$\boldsymbol{J}_{i,j} = \left(\textstyle\sum_k s(e_k)\sigma_{i,k}\sigma_{j,k}\right) \tag{28}$$

where

$$s(e_k) = \frac{1}{\nu + |e_k|}. \tag{29}$$

This correspond to replacing the Gaussian distribution with the following one

$$P(e) = e^{-|e|}\left(1 + \frac{e}{\nu}\right)^{\nu}, \tag{30}$$

which behaves like a Gaussian for $e \ll \nu$ and Laplacian for $e \gg \nu$. The function $s(\cdot)$ checks the confidence with which each measure fits into the least squares procedure and, therefore, can be seen as a representation of the *saliency* of the measure. A common choice for the ν parameter is approximately one standard deviation of the inliers. This approach is known as *iterative reweighted least squares* (IRLS) since s changes on e with each iteration.

IV. UAV VISUAL SERVOING

A visual servoing algorithm can be achieved using the proposed visual tracking system shown in Fig. 4. This algorithm exploits the homogeneous transformation matrix $\boldsymbol{H}^p_c$ of the tracking system within a control law to provide feedback to servo the UAV (for simplicity, we supposed $\mathcal{C}$ coincident with the UAV body frame) to the desired target pose with respect to $\mathcal{P}$. The inverse of the target matrix $\boldsymbol{H}^p_{c,d}$ is computed and the product of this with the current pose yields the transformation from the desired pose to the current pose as $\boldsymbol{H}_e = \boldsymbol{H}^{p\ -1}_c \boldsymbol{H}^p_{c,d}$. The translation velocity vector $\boldsymbol{v}^p_c$ and yaw rate ω_z that must be applied to the UAV are then extracted from this representation as follows

$$\boldsymbol{V}^p_c = \begin{pmatrix} \boldsymbol{S}(\boldsymbol{\omega}^p_c) & \boldsymbol{v}^p_c \\ \boldsymbol{0_{1\times 3}} & 1 \end{pmatrix} = \log(\boldsymbol{H}_e), \tag{31}$$

where $\boldsymbol{S}(\cdot)$ is the skew-symmetric matrix. The corresponding UAV control input can be calculated by multiplying those velocity errors with positive gains. In a real application, a maximum cruise velocity can be imposed through suitable saturation.

V. TEST CASE

blue To demonstrate the performance of the localization algorithm, a simulation test case has been carried out in MATLAB[1], simulating the dynamics of a UAV equipped with a downward camera, flying above a known pipe-rack structure (see Fig. 5). The simulated pipe-rack consists of 5 pipes with the radius varying from 15 and 20 cm. Four of those are parallel, while one tube is placed orthogonally with respect to the parallel ones. In addition, one of the pipe (the first starting from the left) changes its size from 15 to 20 cm. The distance between each pipe is 10 cm. The reference frame has been chosen as described in Section III. In this setup, 25 texture feature points were randomly added to each pipe (100 total), which are 10 m long. However, only a part of these features is visible in the camera's field of view, depending on the position of the UAV during its motion. Along with the simulated pipe-rack, in Fig. 5 is also shown the synthesized

[1]MATLAB 2020b under Windows Operating System has been adopted.

Fig. 5. 3D model of the Pipe rack considered in the MATLAB simulations, with highlighted random-generated texture image features (star points) and visible edges (red/blue lines). The synthesized image corresponding to the 3D scene is also shown where solid (dotted) lines correspond to the actual edges and the starts correspond to the actual texture image points.

Fig. 6. Time histories of the camera position and orientation, UAV linear velocity and orientation velocity.

Fig. 7. Time histories of the position and orientation tracking errors. The orientation data is represented with the ZYX-Euler angles (yaw, pitch, roll).

image corresponding to the implemented 3D scene. The edges of the pipes are samples with 20 tracking points. An ideal camera has been considered, with a frame rate of 25 Hz, a resolution of (720×480)-pixel and the following intrinsic parameters: $f_c = 630$, $r_c = 1$, $s_c = 0$, $u_0 = 360$, and $v_0 = 240$. The controller of UAV has been implemented as presented in [**?**].

In this setup, the UAV is commanded to navigate the pipelines crossing four waypoints. A motion trajectory with continuous acceleration has been planned considering a maximum cruise velocity of $0.5\ m/s$ and $0.2\ rad/s$ for linear and angular velocity, respectively. Taking into account the under-actuation of UAVs commonly used for inspection tasks (i.e. rotary wings), only the rotation around the $Z-$axis (the Yaw, indicated as ψ) is directly controlled. However the full pose of the platform is estimated. The planned position/orientation and relative velocity of the UAV body are depicted in Fig. 6. During the motion, the error between the estimated and actual position and orientation of the UAV has been monitored. To test the algorithm in more realistic conditions, the body of the UAV has been perturbed with external disturbance during the flight to simulate the effect of the wind in forms of wind gusts. In particular, the dryden wind model [**?**] has been used to simulate turbulence effect, by modifying the position calculated by the control action of a quantity proportional to the force and direction of the wind. The effect of such turbulence can be observed in Fig. 6. In particular in the first graph from the top, the commanded/executed trajectory, is possible to see the tracking error caused by the external disturbance injecting a vibration behavior on the flying platform. Moreover, the simulated wind also affects the overall attitude of the UAV, since the low-level controller continuously tries to contrast the wind force.As for the localization results, the position and orientation tracking errors between the estimated position of the UAV and the actual one are reported in Fig 7. These errors are negligible for all motion components during dynamics and depend mainly on the speed of the UAV relative to the camera's frame rate.

To assess the algorithm robustness operating under worse conditions, a second test has been performed assuming a wrong *CAD Model* of the pipe-rack structure and precisely in the knowledge of the distance between the pipelines of the rack is affected by an error. If the actual distance between each

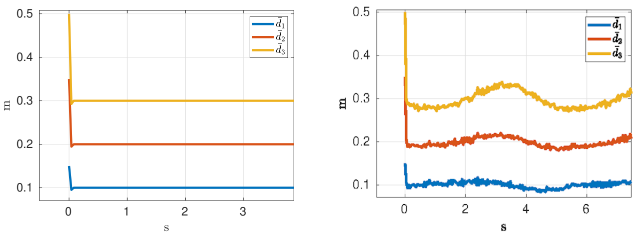

Fig. 8. Time histories of the summed pipe distances $\bar{d}_k$ as described in eq. (24). On the left the ideal case, while on the right with camera quantization error and white Gaussian noise.

pipe has been set at 10 cm (i.e. considering the four pipes of the rack: $\bar{d}_1 = 10$ cm, $\bar{d}_2 = 20$ cm, $\bar{d}_3 = 30$ cm), while the initial estimation has been set to: $\bar{d}_1 = 15$ cm, $\bar{d}_2 = 35$ cm, $\bar{d}_3 = 50$ cm, therefore with a relatively large error. Fig. 8 (left) shows the time history of the estimation of the pipe-rack parameters, which converges to the actual values in few iterations.

blue Finally, a set of simulations has been performed considering different pipelines and extracted visual features. We tested different scenes increasing the number of pipes (from 2 to 5) and the total amount of visual features (from 10 to 50), randomly distributed on the tubes. In these tests, the effects of the quantization of the camera pixels and a withe Gaussian noise with 2 pixel standard deviation has been considered to add measurement noise. For each setup, a total number of 10 simulations have been performed. Results on the mean of the tracking pose error and its standard deviation are reported in Table I. Note that, as expected, the tracking error, in this case, is higher with respect to the first test case due to the pixel error. In addition, it is worth noticing that the number of pipes (and consequently, the extracted edges) affects the orientation error.

VI. Conclusion

In this work, a robust pose-tracking system based on a single camera sensor has been presented to localize a UAV moving on a pipe-rack. A unified framework based on the Lie group SE(3), which allows the simultaneous estimation of the pose of the UAV and some parameters of the pipe-rack model have been described. The tracking algorithm mainly works considering simple image features (i.e. image contours and image features) allowing its execution on low performance computers. In addition, thanks to the computational efficiency of this algorithm, the execution of several iterations of the proposed tracking algorithm could be possible on each acquired image to improve performance. Simulated experiments demonstrate the effectiveness of the proposed approach.

As future approaches, the tracking method will be validated on a real platform operating into a laboratory mock-up. In addition, since the proposed algorithm requires the knowledge of the camera intrinsic parameters, the current work can be extended to estimate such parameters online, during the tracking of the camera motion. Finally, different additional criteria can be added to the formulated reweighted function to improve the robustness of the motion tracking algorithm to

TABLE I
POSITION AND ORIENTATION ERROR CONSIDERING DIFFERENT NUMBER OF PIPES AND VISUAL FEATURES.

f/p	2	3	4	5	
10	0.145±0.02				
1.9±0.5	0.139±0.02				
1.85±0.6	0.13±0.03				
1.2±0.4	0.12±0.01				
1.0±0.3	m				
deg					
20	0.12±0.01				
1.5±0.3	0.124±0.015				
1.55±0.3	0.12±0.02				
1.15±0.4	0.115±0.005				
1.15±0.3	m				
deg					
30	0.11±0.005				
1.2±0.2	0.115±0.001				
1.2±0.2	0.11±0.01				
0.9±0.2	0.112±0.01				
0.8±0.15	m				
deg					
40	0.1±0.01				
0.9±0.1	0.12±0.01				
0.8±0.15	0.09±0.05				
0.9±0.25	0.09±0.01				
0.8±0.1	m				
deg					
50	0.09±0.009				
0.8±0.05	0.085±0.015				
0.79±0.1	0.07±0.02				
0.6±0.15	0.08±0.005				
0.5±0.1	m				
deg					

face wrong localization conditions caused by a wrong visibility of the pipe-rack.

REFERENCES

[1] H. H. Helgesen, F. S. Leira, T. A. Johansen, and T. I. Fossen, “Tracking of marine surface objects from unmanned aerial vehicles with a pan/tilt unit using a thermal camera and optical flow,” in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2016, pp. 107–117.

[2] M. Israel and A. Reinhard, “Detecting nests of lapwing birds with the aid of a small unmanned aerial vehicle with thermal camera,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017, pp. 1199–1207.

[3] T. Drummond and R. Cipolla, “Visual tracking and control using Lie algebras,” in *Proc. Conf. on Computer Vision and Pattern Recognition*, vol. 2, 1999, pp. 652–657.

[4] ——, “Real-time visual tracking of complex structures,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932–946, 2002.

[5] A. Censi and D. Scaramuzza, “Low-latency event-based visual odometry,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 703–710.

[6] W. G. Aguilar, G. A. Rodríguez, L. Álvarez, S. Sandoval, F. Quisaguano, and A. Limaico, “Visual slam with a rgb-d camera on a quadrotor uav using on-board processing,” in *Advances in Computational Intelligence*, I. Rojas, G. Joya, and A. Catala, Eds. Cham: Springer International Publishing, 2017, pp. 596–606.

[7] C. Harris, *Geometry from Visual Motion*. Cambridge, MA, USA: MIT Press, 1993, p. 263–284.

[8] M. Selsis, C. Vieren, and F. Cabestaing, “Automatic tracking and 3D localization of moving objects by active contour models,” in *Proc. of the Intelligent Vehicles*, 1995, pp. 96–100.

[9] C. H. Li and T. I. James Tsay, “Robust visual tracking in cluttered environment using an active contour method,” in *Annual Conf. of the Society of Instrument and Control Engineers of Japan*, 2018, pp. 53–58.

[10] A. Techmer, “Contour-based motion estimation and object tracking for real-time applications,” in *Proc. Int. Conf. on Image Processing*, vol. 3, 2001, pp. 648–651 vol.3.

[11] R. Cipolla and A. Blake, “Image divergence and deformation from closed curves,” *The Int. Journal of Robotics Research*, vol. 16, no. 1, pp. 77–96, 1997.

[12] K. Astrom and F. Kahl, “Motion estimation in image sequences using the deformation of apparent contours,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, no. 2, pp. 114–127, 1999.

[13] G. Klein and T. Drummond, “Robust visual tracking for non-instrumental augmented reality,” in *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2003, pp. 113–122.

[14] A. S. Brandão, F. N. Martins, and H. B. Soneguetti, “A vision-based line following strategy for an autonomous,” in *Int. Conf. on Informatics in Control, Automation and Robotics*, vol. 02, 2015, pp. 314–319.

[15] C. Deng, J. Liu, Y. Liu, and Y. Tan, “Real time autonomous transmission line following system for quadrotor helicopters,” in *Int. Conf. on Smart Grid and Clean Energy Technologies (ICSGCE)*, 2016, pp. 61–64.

[16] H. Xiaoqian, A. Shukla, and H. Karki, “Autonomous ground piipelines tracking via an UAV,” in *Int. Computer Conference on Wavelet Active Media Technology and Information Processing*, 2016, pp. 372–375.

[17] J. Bian, X. Hui, X. Zhao, and M. Tan, “A novel monocular-based navigation approach for autonomous transmission-line inspection,” in *IEEE Int. Conf. on Intelligent Robots and Systems*, 2018, pp. 1–7.

[18] X. Huang, H. Karki, A. Shukla, and X. Zhang, “3D autonomous tracking of buried pipelines via a UAV in a low altitude,” in *IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference*, 2018, pp. 1106–1109.

[19] F. Park, J. Bobrow, and S. Ploen, “A Lie Group formulation of robot dynamics,” *The International Journal of Robotics Research*, vol. 14, no. 6, pp. 609–618, 1995.

[20] T. Drummond and R. Cipolla, “Application of Lie algebras to visual servoing,” *Int. J. Comput. Vision*, vol. 37, no. 1, p. 21–41, Jun. 2000.

[21] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 2010.

[22] E. Karami, S. Prasad, and M. S. Shehata, “Image matching using SIFT, SURF, BRIEF and ORB: Performance comparison for distorted images,” *CoRR*, vol. abs/1710.02726, 2017. [Online]. Available: http://arxiv.org/abs/1710.02726

[23] P. J. Huber and E. M. Ronchetti, *Robust Statistics*, 2nd ed. John Wiley & Sons, 2009.