Received: 14 May 2018 | Revised: 3 December 2018 | Accepted: 3 December 2018

DOI: 10.1002/rob.21859

REGULAR ARTICLE

WILEY

Instance invariant visual servoing framework for part-aware autonomous vehicle inspection using MAVs

Harit Pandya | Ayush Gaud | Gourav Kumar | K. Madhava Krishna

IIIT-Hyderabad, Hyderabad, Telangana, India

Correspondence
Harit Pandya, Robotics Research Center, IIIT-Hyderabad, Gachibowli, Hyderabad 500032, India.
Email: harit.pandya@research.iiit.ac.in

Funding information
TCS research fellowship

Abstract

Visual servoing approaches navigate a robot to the desired pose with respect to a given object using image measurements. As a result, these approaches have several applications in manipulation, navigation and inspection. However, existing visual servoing approaches are instance specific, that is, they control camera motion between two views of the same object. In this paper, we present a framework for visual servoing to a novel object instance. We further employ our framework for the autonomous inspection of vehicles using Micro Aerial Vehicles (MAVs), which is vital for day-to-day maintenance, damage assessment, and merchandising a vehicle. This visual inspection task comprises the MAV visiting the essential parts of the vehicle, for example, wheels, lights, and so forth, to get a closer look at the damages incurred. Existing methods for autonomous inspection could not be extended for vehicles due to the following reasons: First, several existing methods require a 3D model of the structure, which is not available for every vehicle. Second, existing methods require expensive depth sensor for localization and path planning. Third, current approaches do not account for the semantic understanding of the vehicle, which is essential for identifying parts. Our instance invariant visual servoing framework is capable of autonomously navigating to every essential part of a vehicle for inspection and can be initialized from any random pose. To the best our knowledge, this is the first approach demonstrating fully autonomous visual inspection of vehicles using MAVs. We have validated the efficacy of our approach through a series of experiments in simulation and outdoor scenarios.

KEYWORDS
autonomous inspection, instance invariant visual servoing

1 | INTRODUCTION

Visual servoing utilizes image sensory information to move a robotic system toward a goal position with respect to a given object or scene. A visual servoing approach is composed of extracting a set of visual features from image measurements and controlling the robot such that these features match their desired configuration (Chaumette & Hutchinson, 2006). Based on whether the control objective is defined in the Cartesian space or in the image space, the traditional visual servoing approaches are classified into Position-Based Visual Servoing (PBVS) and Image-Based Visual Servoing (IBVS; Hutchinson, Hager, & Corke, 1996). PBVS utilizes visual features to compute the object's pose in the robot's Cartesian space. Estimating the robot's pose requires additional information regarding the geometry of the object. This can be obtained completely by explicit knowledge of the object's 3D model or partially by 3D reconstruction of the object while servoing. IBVS in contrast directly controls the robot in the image space by iteratively minimizing the feature error. However, in IBVS, the controller could lead to local minima as there is no explicit control in the Cartesian space (Chaumette, 1998).

Previous visual servoing approaches used low-level geometric primitives (e.g., corners, lines, contours) as visual features (Chaumette & Hutchinson, 2006) for servoing between two views of the same object as shown in Figure 2a. However, these features do not generalize well for matching between two different object instances due to large appearance and shape variations, as illustrated in Figure 2b. In this paper, we propose part-aware keypoints as visual features that encode the global perspective of the object and retain only the meaningful information in contrast to local appearance-based descriptors, like scale-invariant feature transform (SIFT) (Lowe, 1999) and oriented FAST and rotated BRIEF (ORB) (Rublee, Rabaud, Konolige, & Bradski, 2011), as shown in Figure 2c. To learn these part-aware keypoints, we employ a convolutional neural network (CNN) based on stacked hourglass architecture (Newell, Yang, & Deng, 2016).

Existing visual servoing controllers minimize an error function between the current and the desired configuration of visual features. Since these visual features are geometrically related, a possible solution exists in the SE(3) space such that the error converges to zero at the desired pose. However, when servoing across instances in the same object category, the geometry of the objects may not permit the error to diminish at the desired pose. This limits the scope of traditional visual servoing to views of the same object. Such problems arise frequently in practical scenarios where servoing to a new instance is required, especially in the case of manipulation and navigation. Hence, these approaches are not well suited for performing an autonomous vehicle inspection.

Visual inspection has been widely adopted in industries to assess the quality of the product as well as to identify defects resulted from a manufacturing process. In industries, almost every product goes through the process of visual inspection by trained persons. This procedure of visual inspection is highly monotonous and laborious; therefore, autonomy in visual inspection is rapidly progressing its way into industries. Equipping the autonomous robots with mobile actuation and integrating sensory feedback helps them to perform inspection task outside factory settings. Micro Aerial Vehicles (MAVs) further offer a unique opportunity to achieve sensory measurements from places, which are generally beyond the reach of ground vehicles. Also, they can provide images from a different perspective with finer details. Hence, recently, these MAVs are being used for numerous autonomous inspection tasks, such as for large structures (Bircher, Kamel, Alexis, Oleynikova, & Siegwart, 2016; Morgenthal & Hallermann, 2014), tunnels (Ozaslan et al., 2017), shipboards (Fang et al., 2017), agriculture (Das et al., 2015), and so forth.

These autonomous visual inspection applications commonly require a pipeline comprising exploration and mapping. The common objective is to achieve a dense reconstruction of the object of interest or structure to capture fine details and analyze these details offline. Various aspects of the autonomous inspection problem have been previously studied, such as efficient coverage planning (Bircher, Kamel, Alexis, Burri, et al., 2016), obstacle avoidance (Bircher, Kamel, Alexis, Oleynikova, et al., 2016), map representation (Teixeira & Chli, 2017), vision in degraded environment (Fang et al., 2017), and so forth. Visual inspection of automobiles, in contrast, is slightly different as it is required to visit essential components of the vehicles and decide whether the component or part is in healthy condition or not. To achieve complete autonomy in visual inspection of a vehicle, the MAV should understand the notion of essential components/parts and should navigate to these parts for obtaining a finer view. Visual inspection has several benefits: First, frequent inspection helps monitor the health of the vehicle. Second, this part-aware visual inspection of the vehicle could be helpful in the estimation of damage to the vehicle, especially for an insurance claim. Third, the inspection results could also be used for merchandising the vehicle. As a result, the vehicular inspection is luring numerous ventures, for example, QuickFoto Claim, Express Auto Inspection, and so forth. However, such ventures require the manual effort for collecting a video of the vehicle, which is then processed offline. Car360 Inc.[1] employs either a turntable or a large manipulator for

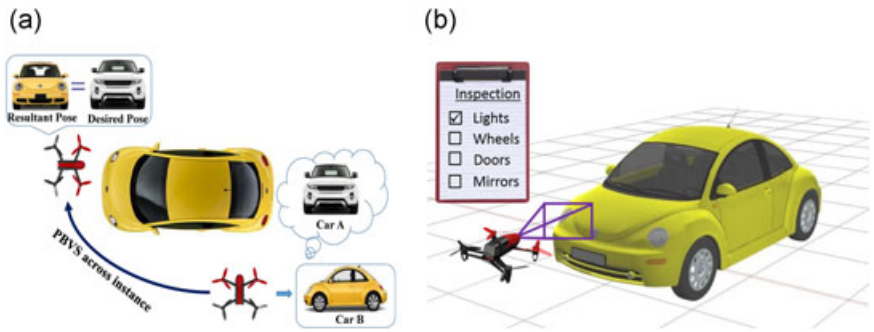

FIGURE 1 Aim: (a) The MAV is initially observing the current instance (yellow Beetle hatchback) from a side view. The instance invariant visual servoing task requires the MAV to move to the front of the Beetle so that observed view matches the desired view of a template instance (white Range Rover). Such navigation maneuver is difficult to attain using existing visual servoing approaches, due to a large variation in shape and appearance between the two instances. (b) Using our instance invariant visual servoing framework, we propose a novel autonomous navigation framework for the visual inspection of a vehicle using a low-cost off-the-shelf MAV. Starting from a random initial pose, the MAV servoes to different parts of the vehicle, like wheels, lights, mirrors, and so forth, to capture finer images of these parts, which plays a crucial role in day-to-day maintenance and damage assessment of the vehicle. MAV: Micro Aerial Vehicle; PBVS: Position-Based Visual Servoing [Color figure can be viewed at wileyonlinelibrary.com]

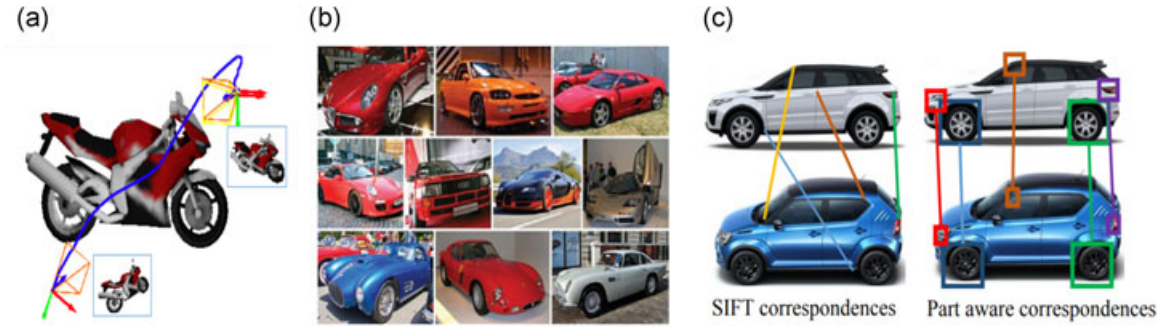

FIGURE 2 (a) IBVS approaches (Crombez et al., 2015) are designed to servo between two views of the same object. (b) Challenges faced by existing IBVS approaches: Variations in appearances, shapes of car, and viewpoints especially nonoverlapping views. (c) Local descriptors, like SIFT-based keypoints, might result in an incorrect matching when the instances are different, whereas our part-aware semantics are more suitable for computing correspondences across object instances. IBVS: Image-Based Visual Servoing [Color figure can be viewed at wileyonlinelibrary.com]

automating the inspection process. However, their setup has scalability issues due to the requirement of a huge infrastructure.

In this study, we address the problem of visual servoing across instances of an object category. Specifically, provided the desired view of an object, the aim is to attain the same desired view for any other instance from that category. Figure 1a describes a servoing scenario, in which an MAV starts from a random initial pose and is required to attain a pose with respect to the current car (yellow Beetle) such that the resultant view matches the desired view of a template instance (white Range Rover). We formulate this problem of servoing across instances as pose induction followed by PBVS, where the pose-induction step infers the desired pose with respect to the current instance from the desired pose of the template. The PBVS step estimates the current pose of the robot with respect to the current instance and moves the robot toward the inferred desired pose. One of the key research challenges in the proposed inspection task is to address the large variations in shape and texture among vehicles. This poses an immense challenge for vision-based classifiers to discriminate them from a background. Not only there are variations in the overall design of vehicles but also the shape and texture of essential parts vary significantly. Detecting the location of these parts is one of the problems, which is gaining interest among computer vision researchers. We propose to employ a stacked hourglass-based CNN for identifying the locations of these essential parts.

After laying the foundation for servoing across instances, we propose a visual navigation pipeline for achieving autonomous part-aware visual inspection of a vehicle using an inexpensive off-the-shelf multirotor MAV equipped with a monocular camera. For the inspection purpose, mere identification of these parts is not sufficient, and the MAV is also required to visit every part for capturing fine details. Thus, we present a novel dilated convolution neural network (DCNN)-based framework that is able to detect the car and assign a dense pixelwise segmentation to these parts. Another key challenge in the proposed work is to navigate the MAV to these parts using a monocular camera and erroneous odometry. We therefore consider this navigation task for autonomous visual inspection as a visual servoing problem, where the objective of MAV is to attain a desired pose with respect to a selected part in the image space (e.g., move the MAV in the Cartesian space such that the selected part should be in the center of image). Hence, here, we propose a hierarchical instance invariant visual servoing pipeline that guides the MAV to acquire the desired pose with respect to every part sequentially. Figure 1b describes a use case of visual inspection, where MAV is moving around a novel instance of a car and is inspecting headlights.

1.1 | Contributions

Our contributions could be summarized as follows:

1. We have introduced a novel problem of instance invariant visual servoing through our previous works (Kumar, Pandya, Gaud, & Krishna, 2017; Pandya, Krishna, & Jawahar, 2015, 2016) that is more suitable in practical scenarios as compared with existing visual servoing approaches for manipulation and exploration tasks. In this paper, we propose a pose-induction framework for visual servoing to a novel object instance. Our framework accommodates the changes in appearance through part-aware keypoints learned using a CNN.
2. To the best of our knowledge, this framework is the first attempt in addressing autonomous inspection of vehicles incorporating semantics. Our approach is able to generalize well despite high intracategory variation in texture and shapes of vehicles. The presented framework is designed to work with a low-cost off-the-shelf MAV.
3. We present a dilated-CNN-based system that is able to achieve state-of-the-art part-segmentation performance with 96.72% pixelwise accuracy and 82.98% intersection over union (IOU) on PASCAL-part data set (X. Chen et al., 2014) for person category.
4. We present multiview data augmentation and optical flow-based Bayesian fusion refinements for improving the segmentation performance, especially for oblique views, which is a common problem when using deep networks due to lack of sufficient

[1]https://www.car360.com

training data. Our approach is able to perform visual servoing across large camera transformations and nonoverlapping scenes, which is a nontrivial task for existing visual servoing approaches.

5. We validated our approach through a series of simulation and field experiments. In the Gazebo simulation, we tested our approach for 12 different vehicles, where the MAV was able to achieve 88.54% accuracy for part inspection. For the field experiments, we evaluated our approach using an off-the-shelf low-cost Parrot Bebop-2 (Parrot SA, Paris, France) MAV, which was controlled using Asus ROG laptop (AsusTek Computer Inc., Taipei, Taiwan, Republic of China) over Wi-Fi. We conducted 16 experiments on seven different vehicles during different illumination conditions, resulting in more than 190 min of completely autonomous flying time. Starting from a random pose, the MAV was able to inspect the required 113 parts out of 128 given parts in total. We further analyzed the performance of our approach on both visual inspection as well as visual servoing parameters, such as the resulting area of a part before inspection, velocity profile, error in visual features, position error, and trajectory of the MAV.

2 | RELATED WORK

2.1 | Shape retrieval and pose estimation from single image

Joint estimation of camera pose and shape of the object from a single image has gained significant attention in computer vision literature due to the availability of economical and compact cameras in comparison to 3D sensors. Early approaches relied on the knowledge of 3D models, for instance, specific reconstruction of the object (Lim, Pirsiavash, & Torralba, 2013). Modern approaches generalize the shape retrieval task to an object category by modeling a given shape as a linear combination of previously selected basis shapes (Hejrati & Ramanan, 2012; X. Zhou, Leonardos, Hu, & Daniilidis, 2017; Zia, Stark, Schiele, & Schindler, 2013) or learned deformable basis shapes (Kar, Tulsiani, Carreira, & Malik, 2015; Vicente, Carreira, Agapito, & Batista, 2014). These weights are then optimized along with the viewpoint to minimize reprojection error. Hejrati and Ramanan (2012) have used alternative shape and pose refinements to solve this optimization problem. Zia et al. (2013) used shape-based priors and maximized the posterior for the pose inference. Recently, X. Zhou et al. (2015) have proposed a convex optimization based on augmented shape and viewpoint basis. However, due to the ill-posed nature of the problem, only reconstruction up-till a similarity transform could be achieved the above-mentioned approaches and full 6D pose could not be recovered. Therefore, these approaches could not be directly used for navigation purposes. Another recent approach proposed by Murthy, Sharma, and Krishna (2017) is able to retrieve full 6D pose of the vehicle relative to the camera, but they assumed the height of the camera to be known. However, in our case, even this approach could not be used due to: (a) Inaccurate odometry, especially large errors in the estimation of MAV's height by the ultrasonic sensor. (b) These approaches require the estimation of six camera parameters and shape parameters; therefore, more they require more keypoints as compared with our approach, which might not be available from all poses especially when the keypoints are predicted by a deep neural network. (c) Even if we are able to retrieve accurate pose, it will be difficult to attain due to erroneous state estimation of the MAV. We, therefore, use a novel pipeline based on instance invariant visual servoing to navigate the MAV to a desired pose.

2.2 | Instance invariant visual servoing

Existing visual servoing approaches servo between two views of the same object. However, in practical scenarios, robotic systems are often required to servo to several objects. This paper addresses the problem of visual servoing across instances of an object category. Specifically, provided the desired view of an object, the aim is to attain the same desired view for any other instance from that category. This study is built upon our previous works on instance invariant visual servoing. In Pandya et al. (2015), we used part-aware keypoints for making the approach robust to textural variation across instances in an object category. We further proposed a linear combination of available 3D models for a servoing iteration. However, the semantic features were computed manually that makes the approach laborious for a large number of object instances. Moreover, the procedure requires a search over all models in all prerendered poses for every visual servoing iteration, which makes the approach computationally expensive. We also proposed a discriminative learning-based framework for visual servoing across instances (Pandya et al., 2016), where we used principal orientation glyph (POG) as visual features and a classification error-based controller for achieving geometry invariance. However, the POG features do not capture the 3D information of the object, which resulted in a relatively smaller convergence domain. In Kumar et al. (2017), we formulated this problem of servoing to a novel instance as pose induction and alignment problem. The pose-induction step infers the desired pose with respect to the current instance from the desired pose of the template. The pose alignment step estimates the current pose with respect to the current instance and moves the robot toward the desired pose using PBVS.

In this paper, we present a hierarchical servoing framework for navigating to essential parts, which extends our pose-induction formulation from Kumar et al. (2017) for navigating the MAV across different sides of the vehicles so that MAV can servo through large camera transformations and even nonoverlapping scenes. We further present an IBVS-refinement step that significantly improves the servoing performance over our previous work (Kumar et al., 2017). After navigating to a side, we propose using Part-IBVS that helps the MAV to servo to essential parts for inspecting them. The evolution of our approach starting from previous versions is presented in Figure 3.

2.3 | Part localization and segmentation

Computing descriptors that provide unique and accurate correspondences among multiple views of the same instance or across different instances have been one of the classical problems in

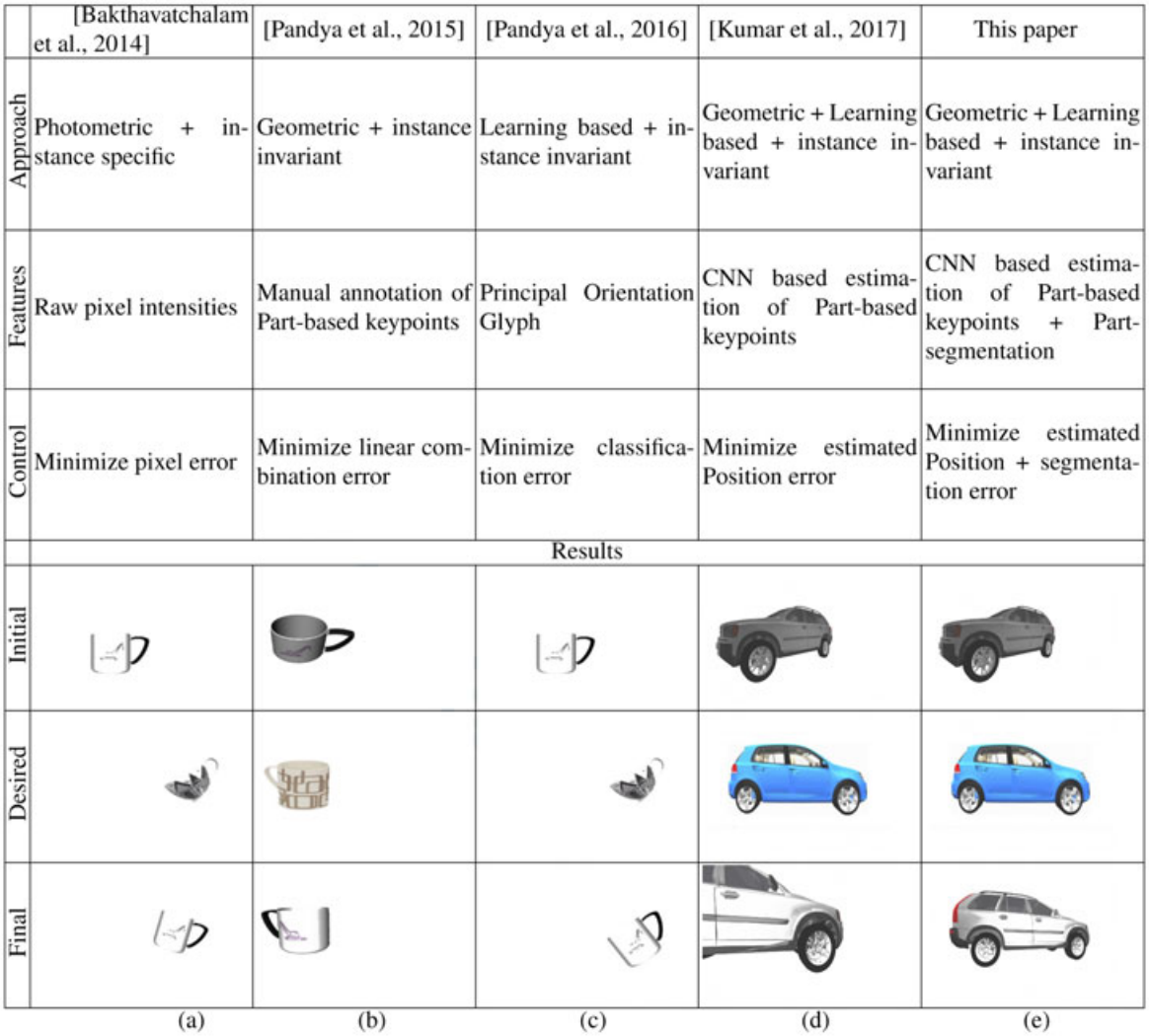

FIGURE 3 Evolution of instance invariant visual servoing. (a) Existing approaches, like photometric visual servoing, attempt to minimize the pixel error without accounting for the variation in shape. (b) Our previous work used a linear combination of 3D models to cater the shape variations. However, a manual annotation of part-based keypoints was required. (c) We then introduced a discriminative learning-based approach, where SVM classification error was used for IBVS. However, the POG features do not capture the 3D information of the object, which limits the convergence basin. (d) We next used CNN for estimating the part-based keypoints. Using the pose-induction-based pipeline, we were able to servo across instances for large camera transformations. (e) In this paper, we extend our previous work by proposing an IBVS-refinement step that significantly improves the visual servoing performance. CNN: convolutional neural network; IBVS: Image-Based Visual Servoing; POG: principal orientation glyph; SVM: support vector machine [Color figure can be viewed at wileyonlinelibrary.com]

computer vision. Previous approaches from computer vision literature report superior performance in keypoint correspondences when the keypoints are conditioned on object category, especially when the keypoints were semantically related to object's parts (Felzenszwalb, Girshick, McAllester, & Ramanan, 2010; Maji & Shakhnarovich, 2012). Motivated from recent breakthroughs in CNNs, Tulsiani and Malik (2015) presented a CNN that was able to learn part-aware keypoints through supervision. They reported a significant improvement in keypoint prediction accuracy by conditioning keypoints inference on viewpoint estimations. Recently, Newell et al. (2016) proposed a stacked hourglass architecture for CNN that showcased state-of-the-art results for keypoint prediction for the human category. In this paper, we use a similar stacked hourglass CNN from our previous work (Kumar et al., 2017) trained on PASCAL 3D data set (Xiang, Mottaghi, & Savarese, 2014) for cars. We obtained superior results for keypoints detection over Tulsiani and Malik (2015) for 'car' object category.

As compared with object localization where only the center of the object is estimated, the semantic segmentation problem consists of assigning a class label to every image pixel. Thus, the segmentation helps in better understanding about the object and the scene compared with mere object localization. Classical segmentation approaches based on Markov Random Fields (Blake,

Rother, Brown, Perez, & Torr, 2004; Tang, Gorelick, Veksler, & Boykov, 2013) considered this as graph labeling problem, where the objective was to minimize the network's energy by assigning correct labels to every pixel. Another branch of approaches based on Conditional Random Fields (CRFs; Gonfaus et al., 2010; Maire, Stella, & Perona, 2011; Plath, Toussaint, & Nakajima, 2009) used multiple cues from different modalities to improve segmentation performances. Recent advances in CNNs have revolutionized this problem of segmentation. In the past couple of years, several CNN-based models were proposed for category-level segmentation (Badrinarayanan, Kendall, & Cipolla, 2015; L.-C. Chen, Papandreou, Kokkinos, Murphy, & Yuille, 2016; Dai, He, & Sun, 2016; Yu & Koltun, 2015). For the purpose of visual inspection of a vehicle, we are more interested in part-level segmentation. A few recent approaches cater use CNNs for part-level segmentation (Oliveira, Bollen, Burgard, & Brox, 2018; B. Zhou et al., 2017).

In this paper, we present a dilated-CNN-based framework trained on PASCAL-parts data set (X. Chen et al., 2014) for segmenting essential parts of the vehicle. Since our control framework employs this part-segmentation feedback for visual servoing, we are required to see cars from several viewpoints. However, the training data available from PASCAL-parts data set do not provide an extensive set of viewpoints. Therefore, we present a simulation framework for synthetic data augmentation. Oliveira et al. (2018) have also reported that data augmentation (spatial and color) results in significant performance improvement. Furthermore, in our case, the camera attached to the robot is also moving; therefore, we use this active vision to our advantage by fusing segmentations from multiple views. We propose optical flow-based warping along with the Bayesian fusion (Ma, Stückler, Kerl, & Cremers, 2017), for improving the segmentation performance.

2.4 | Autonomous visual inspection

Early works on autonomous visual inspection using a fixed camera were tailored to the requirement of determining the quality of the product during a manufacturing process (Chin & Harlow, 1982). With the addition of mobility to the camera, the applications of autonomous visual inspection increased by many folds. Recently, the MAVs are being extensively used for inspection of large structures (Bircher, Kamel, Alexis, Oleynikova, et al., 2016; Morgenthal & Hallermann, 2014), tunnels (Ozaslan et al., 2017), shipboards (Fang et al., 2017), agriculture (Das et al., 2015), and so forth, due to the ability of MAVs flying stably at lower altitudes and reaching places that are difficult to approach for humans. To achieve complete autonomy in inspection problems, challenges related to navigation need to be addressed, such as accurate state estimation. Several navigation approaches (Bircher, Kamel, Alexis, Oleynikova, et al., 2016; Dryanovski, Valenti, & Xiao, 2013; Shen, Michael, & Kumar, 2011; Usenko, von Stumberg, Pangercic, & Cremers, 2017) rely on accurate depth sensors, like LIDARs, for accurate state estimation and mapping, which makes the MAV heavy and expensive. Fraundorfer et al. (2012) and Schauwecker and Zell (2014) in contrast use a stereo camera pair for depth estimation, which requires matching features between frames and increases time delay in the system. Although the stereo pair is not heavy compared with the LIDAR sensor, however, it is still expensive and is not present in the majority of off-the-shelf MAVs, such as Parrot's Bebop (Parrot SA), DJI Phantom (SZ DJI Technology Co., Ltd. Shenzhen, Guangdong, China), and so forth. Recent approaches from Lin et al. (2018), Weiss, Scaramuzza, and Siegwart (2011), and Wu, Johnson, Kaess, Dellaert, and Chowdhary (2013) propose a visual inertial navigation framework that fuses the inertial measurement unit (IMU) measurements with image measurements from monocular camera, but for accurate localization their framework requires a global shutter camera and hardware synchronization between the clocks of camera and IMU, which is again not present in off-the-shelf MAVs. Hence, in this paper, we use visual servoing in the image space to navigate the MAV to the desired pose of essential parts circumventing the requirement for accurate pose estimation.

3 | PROBLEM DESCRIPTION

The problem of vehicular inspection, as it is considered in this paper, consists of a low-cost off-the-shelf MAV equipped with a monocular camera for inspecting a given vehicle (current instance). The given vehicle could be any novel instance of a car, previously unseen by our approach. We, therefore, formulate this navigation problem as an instance invariant visual servoing, where a template model is used to estimate the pose of the MAV with respect to the given vehicle and navigate the MAV to sides (left $\mathcal{F}_l^*$, right $\mathcal{F}_r^*$, and front $\mathcal{F}_f^*$) of the vehicle from a random initial pose $\mathcal{F}_0$. Here, the images of sides of the template model act as desired views, in which the MAV is required to servo sequentially.

The objective of the inspection problem is to visit the pose $\mathcal{F}_i^* = [X_i^*, Y_i^*, Z_i^*, \theta_i^*], \forall\ i \in N_P$ corresponding to every essential part i among the set of all parts N_P of the vehicle (wheels, headlights, and mirrors) starting from a random initial pose $\mathcal{F}_0$. We define $\mathcal{F}_i^*$ such that the part i occupies a predefined area A_i^* and lies in the center of the image captured by MAV at $\mathcal{F}_i^*$, that is, $[u_i, v_i] = [N_i/2, M_i/2]$, for the image with size $M_i \times N_i$. We assume that the navigable workspace is obstacle free and the initial pose of the MAV, denoted by $\mathcal{F}_0$, has the vehicle in the field-of-view of the camera. All the computations need to be carried out near real time.

4 | OVERALL PIPELINE

The overall pipeline comprises a perception and a control module. The perception module processes the image sequence and returns (i) a bounding box capturing the car, (ii) keypoint locations of the parts in the image, and (iii) pixelwise segmentation of essential

parts. The control pipeline is responsible for navigating the MAV to the desired pose $\mathcal{F}_i^*$ for every part in a sequential order. The MAV starts at a random pose $\mathcal{F}_0$ and visits a vantage pose for every side $\mathcal{F}_s^*$, $\forall\ s \in \{l, f, r\}$ (front $\mathcal{F}_f^*$, left $\mathcal{F}_l^*$, and right $\mathcal{F}_r^*$). After reaching to the vantage point, the MAV visits all the parts of that side sequentially. The key challenge here is that the poses $\mathcal{F}_i^*, \mathcal{F}_s^*$ are not known in the Cartesian space, rather these are to be inferred from images using the vision pipeline. As shown in Figure 4a, we divide the control pipeline into three modules: Pose induction, side navigation and part servoing, where the pose-induction and side-navigation modules are the components of the instance invariant visual servoing framework. The outputs of every module are visualized in Figure 4b. We now briefly summarize the functionality of every module.

The pose-induction module is responsible for reconstructing the given vehicle (current instance) and estimating the vantage poses for every side $\mathcal{F}_s^*$. Provided a random starting pose $\mathcal{F}_0$, the MAV initially follows a straight line maneuver orthogonal to the optical axis for reaching an arbitrary pose $\mathcal{F}_0'$, such that the corresponding images I_0 and I_0' captured from $\mathcal{F}_0$ and $\mathcal{F}_0'$ form a multiview stereo pair. We refer to this step as multiview stereo initialization. We further use our keypoint prediction network to extract the part locations from these images. These part locations from I_0 and I_0' are then used for triangulation and semantic reconstruction of parts as 3D keypoints $P_i = [X_i, Y_i, Z_i]$ in the Cartesian space. A standard model (template) of a car is then aligned with the reconstructed 3D model so that the vantage poses $\mathcal{F}_s^*$ could be transferred from the template model to the current reconstruction, as for the template model these vantage poses can be easily computed using 2D–3D correspondences from desired images. As shown in Figure 4b, the pose-induction module produces four outputs: (a) Semantic reconstruction of the current instance, (b) initial pose of MAV with respect to the current instance $\mathcal{F}_0$, (c) desired poses of all sides $\mathcal{F}_s^*$, $\forall\ s \in \{l, f, r\}$, and (d) the order in which these sides should be visited.

After $\mathcal{F}_s^*$ is estimated in the Cartesian space, the side-navigation module guides the MAV to $\mathcal{F}_s^*$ using PBVS. Here, we use odometry provided by the MAV for localization. However, the

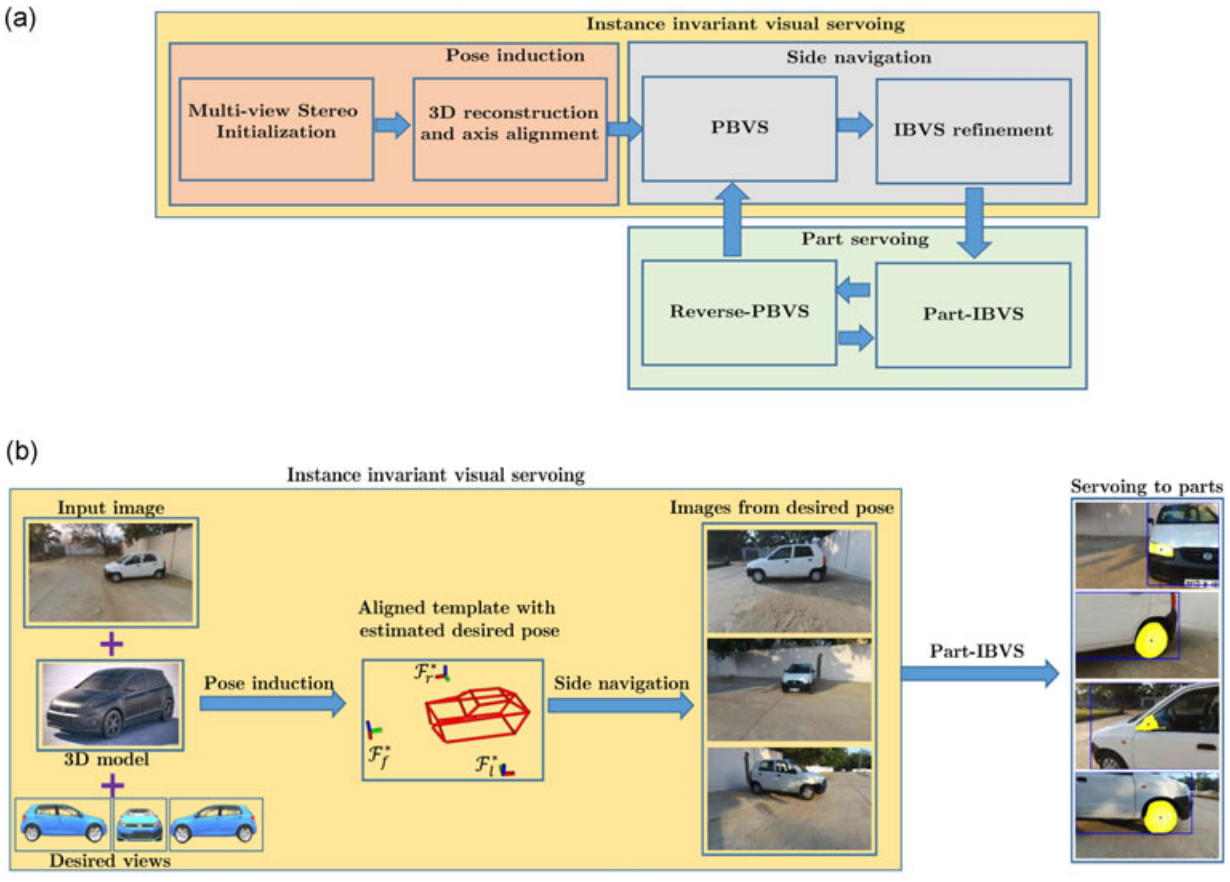

FIGURE 4 Overall workflow. (a) The overall pipeline of the proposed approach and (b) the results of major components of the pipeline for a field test case. The multiview stereo initialization, based on keypoints encoding the part locations, gives a semantic reconstruction of the vehicle. This is aligned with the template of a car. The pose-induction module is also responsible for estimating the desired pose in the Cartesian coordinates. The side-navigation module assigns an ordering to these poses and helps the MAV to visit every visit every facet/side (left, right, front, or back). This is followed by part servoing for zooming-in to every part. IBVS: Image-Based Visual Servoing; MAV: Micro Aerial Vehicle; PBVS: Position-Based Visual Servoing [Color figure can be viewed at wileyonlinelibrary.com]

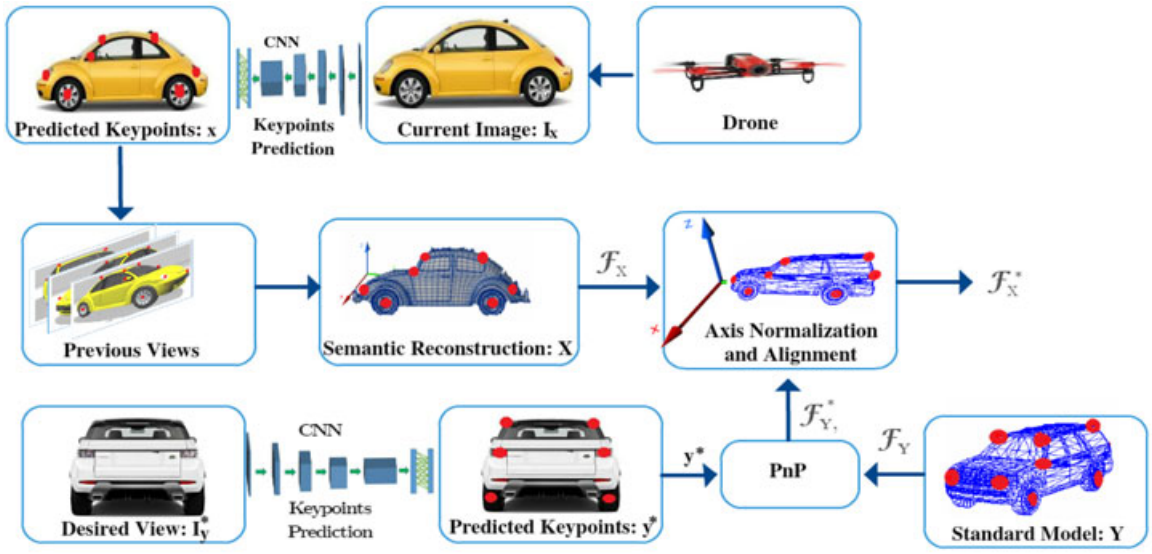

FIGURE 5 Pose induction: The objective of the pose-induction module is to estimate a vantage pose for the current side $\mathcal{F}_s^*$. This is achieved by using a standard 3D model of a car (template) and the provided desired view in the form of an image. Our deep network predicts the keypoints, which, along with template instance model information, is used to predict the desired transformations $\mathcal{F}_Y^*$ in its canonical frame using perspective-n-point (PnP) solver. The keypoints predicted on the current image along with the previous views are used for semantic reconstruction of the current object X and the origin $\mathcal{F}_X^*$, which is its centroid. This reconstruction is aligned with $\mathcal{F}_Y^*$ using the axis normalization and alignment module in a canonical frame $\mathcal{F}_v$ to obtain the desired pose specific to the current vehicle $\mathcal{F}_s^* = \mathcal{F}_X^*$. This pose is then fed to the side-navigation module that employs position-based visual servoing (PBVS) controller for generating control commands for the MAV. CNN: convolutional neural network; MAV: Micro Aerial Vehicle [Color figure can be viewed at wileyonlinelibrary.com]

odometry measurements obtained by MAV are inaccurate and tend to drift over time; therefore, we additionally use an image moment-based visual servoing for pose refinement. The resulting of images captured by the MAV after the refinements is also shown in Figure 4b.

Eventually, our part-servoing module guides the MAV to a desired pose $\mathcal{F}_i^*$ for all parts belonging to that side in a sequential order. We employ the semantic part segmentation parameterized by image moments $p_i = [u_i, v_i, A_i]$ as visual features for navigating to parts using IBVS. After attaining the desired pose, we use the reverse-PBVS module to return to $\mathcal{F}_s^*$ and continue this procedure until all the parts visible from that side are servoed. Then, to switch side, we again use PBVS followed by IBVS refinement to attain the vantage point for the next side ($\mathcal{F}_l^*$ or $\mathcal{F}_r^*$ or $\mathcal{F}_f^*$). This process is repeated for all parts and is summarized in Algorithm 1. Images along with overlaid segmentations captured by the MAV from a few $\mathcal{F}_i^*$ are shown in Figure 4b for a test case. We also encourage the readers to refer the video at the project website[2] to see our workflow in action.

It should be noted that the current instance of the car is different from the standard model (template) of the car, thus classical IBVS and PBVS methods could not be directly used. Also, we need to navigate to all sides of the car that requires servoing between nonoverlapping scenes, which could not be achieved only by using IBVS, thus we employ this switching-based hierarchical servoing approach.

[2]Project website: http://robotics.iiit.ac.in/people/harit.pandya/vi_inspection

Algorithm 1: Instance invariant visual servoing for visual inspection

```
Multi-view stereo-initialization maneuver
Semantic 3D Reconstruction
Alignment with standard model and estimate F_s^*
forall sides do
    Identify side and plan side-visit ordering
    PBVS to F_s^*
    IBVS refinement
    forall parts of current side do
        Part-IBVS
        PBVS back to F_s^*
    end
end
```

5 | POSE INDUCTION

The objective of the pose-induction module is to estimate the desired pose $\mathcal{F}_s^*$ using a template model Y along with its desired image I_s^* for the current side and transfer it to the current instance X. We employ a CNN to compute the part correspondences x from current I_X and desired image $I_Y^* = I_s^*$ of the current side. Once we obtain the predictions y^* from I_Y^*, we use its 3D model (template) to compute the desired camera pose $\mathcal{F}_Y^*$ by solving the perspective-n-point (*PnP*) problem, as described in Figure 5. Kindly note that here we are dealing with two reference frames, one frame is attached to the center of current object instance, which is being servoed $\mathcal{F}_X$ and other is attached to the template $\mathcal{F}_Y$. Hence, we align both X and Y in a single virtual canonical frame $\mathcal{F}_v$, so that $\mathcal{F}_s^* = \mathcal{F}_X^*$ could be transferred from the template $\mathcal{F}_Y^*$. Also, note that the $\mathcal{F}_Y^*$

computation is required only once per side and hence could be performed offline. Our framework performs real-time reconstruction and servoing of the current instance.

5.1 | Keypoint prediction

In this study, we leveraged the recent hourglass architecture-based deep CNN for our task of part-aware keypoint prediction. This network architecture was initially proposed by Newell et al. (2016) for human pose estimation. The network architecture comprises eight hourglass modules stacked one after the other. Each hourglass is an encoder–decoder setup and is followed by residual module (we encourage readers to refer Newell et al., 2016 for complete network architecture). We have trained this network on annotated images of cars from PASCAL 3D data set (Xiang et al., 2014) using Stochastic Gradient Descent (SGD) with a Euclidean loss. The design of hourglass network captures information at multiple scales similar to Tulsiani and Malik (2015). However, this model is faster and more accurate compared with Tulsiani and Malik (2015) due to the following architectural improvements: (a) Use of stacked hourglass provides an end-to-end solution for estimating part-aware keypoints, which is more optimal than ensemble of three networks, since all parameters are jointly optimized rather than an inference over independently optimized parameters. (b) The stacking of hourglass modules assures both repeated bottom-up and top-down re-evaluation of initial feature estimates. (c) We trained the network to estimate the confidence scores along with the image coordinates corresponding to each keypoint for a given image. The low scores corresponding to occluded parts or less guessable parts helped us to filter out the less confident predictions by the network. These predictions were further used as features for reconstruction, pose estimation while visual servoing. We evaluated this network on PASCAL 3D data set for vehicles, and our network is able to outperform previous approaches as shown in Section 8.1.

5.2 | Desired pose estimation

The problem of determining pose of a calibrated camera from n correspondences between 3D reference points and their 2D projections is known as *PnP problem*. The solution to this *PnP* problem is used for estimating the camera extrinsics $(\mathbf{R}, \mathbf{t})$. Provided a set of n points X_i in 3D and their 2D correspondences x_i, estimating the camera pose can be posed as a problem of minimizing reprojection error as

$$\min_{\mathbf{R},t} \sum_{i=1}^{n} ||K(\mathbf{R}X_i + t) - x_i||^2 \quad \text{subject to: } \mathbf{R}^T\mathbf{R} = I. \tag{1}$$

PnP is a well-established problem in a 3D geometry, and there are multiple solutions to the problem. We have used *accurate and scalable solution to the perspective-n-point* (*ASPnP*) (Zheng, Sugimoto, & Okutomi, 2013) since it solves the 3D–2D correspondences using a Gröbner basis solver, which guarantees a globally optimal camera pose. The desired pose of the camera $\mathcal{F}_Y^*$ is determined from the annotated 3D model Y, and the desired 2D part correspondences y^* using *PnP*.

5.3 | Semantic reconstruction

In our approach, the features used for reconstruction are keypoints, which uniquely corresponds to parts of a vehicle; therefore, we use the term semantic reconstruction. The process of semantic reconstruction starts with a multiview stereo initialization, that is, giving a translation orthogonal to the optical axis of the camera in a horizontal plane to form a stereo image pair. Assuming the knowledge of camera parameters, the stereo pair obtained is used in triangulation for estimating 3D coordinates of the visible keypoints. The triangulated points are determined in the initial camera frame $\mathcal{F}_0$, which is then transferred to a frame $\mathcal{F}_X$ that is attached to the centroid of the reconstructed model X. The odometry readings are used for determining the actual scale of the current instance, which is further utilized in the pose alignment step to scale up the normalized reconstruction of the current instance. Note that only a few keypoints are visible from the initial pose, hence only a partial 3D model is reconstructed.

5.4 | Normalization and alignment to the canonical frame

The reconstruction of the current instance X from semantic reconstruction pipeline is in the frame $\mathcal{F}_X$, whereas the desired camera pose given by PnP is in the frame of template instance $\mathcal{F}_Y$. Therefore, to compute the desired camera pose with respect to X, we need the transformation between $\mathcal{F}_X$ and $\mathcal{F}_Y$, which unfortunately is not always available. We tackle this issue by defining a canonical frame $\mathcal{F}_v$ and transforming (aligning) both $\mathcal{F}_X$ and $\mathcal{F}_Y$ to $\mathcal{F}_v$ using an alignment protocol. A valid alignment protocol requires exactly three rules: one rule to define an origin and two rules to define the alignment of any two coordinate axes. As an example, consider the alignment between the partial reconstructions of two car instances as shown in Figure 6a, where a current instance given by the blue wireframe is required to align with a template instance given by the red wireframe. Further, consider a set of four keypoints corresponding to "left front wheel," "right front wheel," "left rear wheel," and "right rear wheel." We select the origin of canonical frame $(\mathcal{F}_v)$ as the right front wheel. Thus, the origin of the template instance $(\mathcal{F}_Y)$ is also selected at right front wheel. We place the frame attached to vehicle $\mathcal{F}_X$ at the centroid of the reconstructed points, which is at known transformation from our selected origin (right front wheel). The x-axes in all the frames are parallel to the ray from the right front wheel to the left front wheel, and the y-axes are parallel to the ray from the right front wheel to the right rear wheel starting at their corresponding origin as shown in Figure 6b. By aligning the frame attached to the template and the canonical frame, we get the transformations ${}^v\mathbf{T}_Y$, whereas ${}^v\mathbf{T}_X$ is known since both $\mathcal{F}_X$ and $\mathcal{F}_v$ belong to the same instance X. To maintain the homogeneity in the scale, all the keypoints of the template instance have to be in the

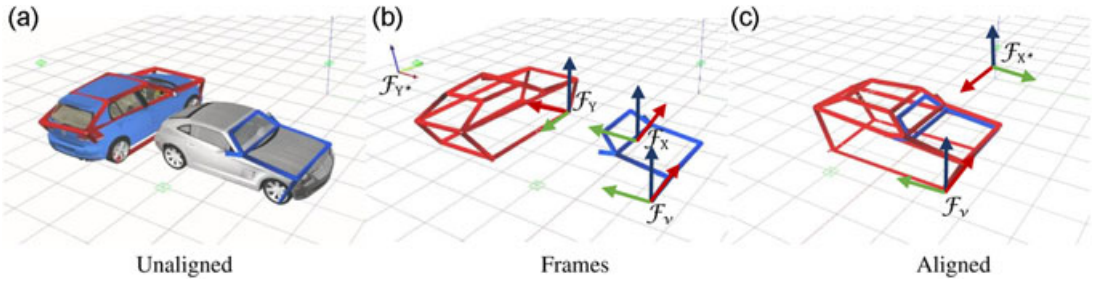

FIGURE 6 Axis alignment: The axis alignment is one of the crucial steps for the pose-induction module. This procedure transfers the desired pose $\mathcal{F}_{Y^*}$ from the template instance Y to the current instance X. (a) The partial reconstruction of the current instance X and the template instance Y are shown with blue and red wireframes, respectively. (b) All the required frames are shown. $\mathcal{F}_{Y^*}$ is the desired camera pose obtained by using PnP between desired image and template model (red wireframe). The frame $\mathcal{F}_X$ is attached to the centroid of the current instance. We then select a suitable keypoint (say left front wheel) and attach the canonical frame $\mathcal{F}_v$ capturing a semantic relation of keypoints (such as x-axis of the frame is parallel to line joining the front wheels). Next, we select the same keypoint capturing the same semantic relation in template instance and attach the template frame $\mathcal{F}_Y$. (c) Since $\mathcal{F}_Y$ and $\mathcal{F}_v$ are semantically the same frame, therefore, they could be directly aligned and using Equation (2), we can compute the desired pose in current frame $\mathcal{F}_{X^*}$. Note that axis alignment is only possible since our part-based keypoints have semantic meaning unlike SIFT or ORB keypoints. PnP: perspective-n-point [Color figure can be viewed at wileyonlinelibrary.com]

same scale as that of the current reconstruction. Since the model is reconstructed partially, so we assign the scale between two keypoints of the longest visible side to all the keypoints. This assigns an approximate scale to the template, which results in some alignment errors. The error in alignment can be reduced by using a linear combination of more than one templates similar to Pandya et al. (2015) and X. Zhou et al. (2015). However, we do not focus on this objective because, even if the correct pose is estimated, it would be difficult to attain it due to incorrect odometry estimates provided by the MAV. Therefore, in this study, we employ an IBVS-based refinement after alignment and PBVS step as explained in Section 6.1.

The pose-induction process can then be explained with the following equation:

$$\mathcal{F}_X^* = ({}^v\mathbf{T}_X)^{-1} \; {}^v\mathbf{T}_Y \mathcal{F}_Y^*, \tag{2}$$

where $\mathbf{T}$'s are 4×4 homogeneous transformation matrices and could be represented as

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}.$$

After alignment, the estimated desired pose for the current side $\mathcal{F}_s^* = \mathcal{F}_X^*$ is used by the PBVS controller for navigating the MAV from the initial pose $\mathcal{F}_0$ to the current side $\mathcal{F}_s^*$, since now they are defined in the same frame $\mathcal{F}_X$. This alignment is only possible because our keypoints have a semantic meaning associated with them.

6 | PBVS TOWARD SIDE

PBVS approaches control the robot directly in the Cartesian space, and thus the resulting camera trajectory is an optimal path, that is, a straight line. Furthermore, for PBVS approaches, the interaction matrix is full rank; therefore, these approaches can tackle large camera variations without getting stuck in local minima. These properties of PBVS approaches make them ideal candidate for visual navigation tasks especially between nonoverlapping scenes. In our problem, we need to navigate the MAV from $\mathcal{F}_0$ to all the sides of the vehicle sequentially, starting with the current side ($\mathcal{F}_s^* \in [\mathcal{F}_r^*, \mathcal{F}_l^*, \mathcal{F}_f^*]$). However, we only know $\mathcal{F}_r^*$ or $\mathcal{F}_l^*$ or $\mathcal{F}_f^*$ with respect to the template Y. Thus, the standard PBVS techniques could not be directly applied here.

Therefore, the pose-induction step as explained in Section 5 employs a template model Y and the semantic reconstruction X to estimate the required camera pose $\mathcal{F}_s^*$ such that the camera projection at $\mathcal{F}_s^*$ matches I_s^*. As a result, now both $\mathcal{F}_0$ and $\mathcal{F}_s^*$ are in the same frame. Therefore, we can apply classical PBVS to navigate the MAV to $\mathcal{F}_s^*$. Similar to the classical PBVS, we consider $\mathcal{F}_c$ as the current pose and $\mathcal{F}_s^*$ as the desired pose, where the current pose of the MAV is obtained from the odometry provided by MAV. Let ${}^{s^*}\mathbf{R}_c$ and ${}^{s^*}\mathbf{t}_c$ denote the rotation and translation of the current pose of MAV $\mathcal{F}_c$ with respect to the desired pose $\mathcal{F}_s^*$. Then, PBVS control law could be stated as follows (Chaumette & Hutchinson, 2006):

$$\mathbf{v}_c = -\lambda_l \; {}^{s^*}\mathbf{R}_c^T \mathbf{t}_c, \tag{3}$$

$$\omega_c = -\lambda_a \theta \mathbf{u}, \tag{4}$$

where $\mathbf{v}_c$ and ω_c are linear and angular commanded velocities, and λ_l and λ_a are step sizes corresponding to linear and angular velocities, respectively. $\theta\mathbf{u}$ is the angle-axis representation of ${}^{s^*}\mathbf{R}_c$.

6.1 | IBVS-based refinement

The PBVS controller explained above generates the required velocity command to move the MAV from $\mathcal{F}_0$ to $\mathcal{F}_s^*$. However, the PBVS controller requires the estimation of the current pose of the camera

at every time instance, which was furnished by noisy odometry reading of the MAV. Moreover, there could also be errors on account of the large difference in shape between the current instance and the template. Hence, MAV could attain an incorrect desired pose $\mathcal{F}_s^{'} = \mathcal{F}_s^* + {}^{s'}\mathbf{T}_{s^*}$. We, therefore, use an IBVS refinement to minimize the error ${}^{s'}\mathbf{T}_{s^*}$ in the desired pose.

For this IBVS-refinement step, we use the segmentation mask generated by our part-segmentation module (refer to Section 7.1) and extract image moments (center and area of segmentation). These image moments $[x_g, y_g, A]$ are used as visual features to control the MAV using IBVS. The approach is similar to Part-IBVS, and the details are described in Section 7.2. In contrary to Part-IBVS, here we use the segmentation of the entire vehicle for extracting the image moments. Note that the desired image ($I_s^* = I_Y^*$) remains the same as that for the PBVS module.

Note that while servoing between sides, the vehicle could itself become an obstacle we circumvent this issue by (a) ensuring that the MAV always servos to an adjacent side and not directly to the opposite side of the vehicle. (b) Taking a conservative approach, we assign intermediate waypoints while performing IBVS. These waypoints are selected as images of corners of the vehicle, so that the MAV follows an elliptical trajectory around the vehicle instead of a straight line, which avoids collision with the vehicle.

7 | IBVS TOWARD PARTS

The instance invariant PBVS presented in Section 6 navigates the MAV to a side of the vehicle $\mathcal{F}_s$ using part-based keypoints. Approaching further to a part requires zooming-in to a single keypoint, which is a degenerate scenario for IBVS as the interaction matrix becomes singular and hence noninvertible. Also, PBVS could not be used due to erroneous odometry, which could lead to fatal consequences as the MAV is closer to the vehicle. Therefore, we propose to employ image moments of the part-segmentation mask as visual features for zooming-in into a part. We refer to this as Part-IBVS. The moments used are the location of the centroid of the part-segmentation mask (x_g, y_g) and its area A. We propose a novel cascaded DCNN architecture for computing the semantic segmentation of a part. We further improve the performance by using synthetic data augmentation and multiview segmentation. We describe the existing approaches for part segmentation in Section 7.1 and present the cascaded architecture in Section 7.1.1, and the necessary refinements to adapt the network for visual servoing are described in Sections 7.1.2 and 7.1.3. Implementation details regarding the data set, network training, and evaluation metrics are explained in Sections 7.1.4–7.1.6. The part-segmentation masks produced by the network are further used in extracting the visual features and controlling the MAV using IBVS as described in Section 7.2.

7.1 | Part segmentation

Recent CNN-based approaches (Badrinarayanan et al., 2015; Dai et al., 2016; B. Zhou et al., 2017) define the problem of semantic segmentation as assigning every pixel i of input image I_x, a class label y_i from a list of predefined categories N_{cl}, which is obtained by training the network under supervised settings on a data set of N_{tr} samples, each consisting of an image $I_{x_{tr}}$ and corresponding ground truth label provided in the form of pixelwise annotation image $I_{y_{tr}}$. The network weights θ for the architecture f are learned by minimizing the following cost function:

$$\theta = \arg\min_{\theta} \sum_{i=1}^{N_{tr}} \sum_{k=1}^{N_{cl}} \log(y_i)\, \mathrm{smax}(f_k(I_{x_i}; \theta)), \tag{5}$$

where smax denotes the soft-max function,

$$\mathrm{smax}(m_k) = \frac{\exp(m_k)}{\sum_{l=1}^{N_{cl}} \exp(m_l)} \tag{6}$$

over all classes. The inference step consists of assigning a label k with the maximum score to each pixel i of the input image I_x:

$$y(i) = \max_k (f_k(x; \theta)). \tag{7}$$

In this paper, we propose a cascaded DCNN-based framework for part segmentation of a given vehicle. Previously, DCNN was utilized by Yu and Koltun (2015), for semantic segmentation to improve the performance over fully convolutional networks (Long, Shelhamer, & Darrell, 2015). Yu and Koltun (2015) dropped pool4 and pool5 from fully convolutional VGG-16 network (Simonyan & Zisserman, 2014) and replaced the following convolutions with dilated convolutions, which improved the segmentation accuracy. B. Zhou et al. (2017) proposed a branched cascaded architecture for semantic segmentation. They divide the image into three branches, (a) objects, (b) stuff, and (c) parts, and use three different streams for each of them. The loss function is constructed by combining the individual loss of each of the branches. Similarly, the network proposed by Oliveira et al. (2018) consists of two streams: one for object detection and other for the parts. The part network operates at twice the scale of the object similar to Felzenszwalb et al. (2010), to capture finer details of the parts. Dai et al. (2016) also indicated that capturing parts conditioned over the object masks improve the performance of parts segmentation Table 1.

7.1.1 | Two-stage stage cascading

In this paper, we present a two-stage cascaded DCNN network architecture for object part segmentation as shown in Figure 7. In contrast to Oliveira et al. (2018) and B. Zhou et al. (2017), we propose to sequentially stack the two cascaded modules, and the motivation behind this is to condition the part segmentation on object masks. The first DCNN module (DCNN module1) in the cascade is responsible for object segmentation at instance level. A bounding box is then computed from the segmentation mask produced by DCNN module1 for every object instance in the image. The object instance is then zoomed-in based on the bounding box produced by DCNN module1, which is then fed to the second module of the cascade (DCNN module2) after resizing. Thus, DCNN module2 processes every instance separately at the same scale for estimating

TABLE 1 Network architecture describing individual module of the stacked DCNN used for pixelwise semantic segmentation

Layer name	Kernel size	Pad	Stride	Dilation	Output size
Input	–	–	–	–	1 × 3 × 376 × 376
conv1_1	3 × 3	1	1	1	1 × 64 × 376 × 376
conv1_2	3 × 3	1	1	1	1 × 64 × 376 × 376
pool1	2 × 2	0	2		1 × 64 × 188 × 188
conv2_1	3 × 3	1	1	1	1 × 128 × 188 × 188
conv2_2	3 × 3	1	1	1	1 × 128 × 188 × 188
pool2	2 × 2	0	2		1 × 128 × 94 × 94
conv3_1	3 × 3	1	1	1	1 × 256 × 94 × 94
conv3_2	3 × 3	1	1	1	1 × 256 × 94 × 94
conv3_3	3 × 3	1	1	1	1 × 256 × 94 × 94
pool3	2 × 2	1	1		1 × 256 × 47 × 47
conv4_1	3 × 3	1	1	1	1 × 512 × 47 × 47
conv4_2	3 × 3	1	1	1	1 × 512 × 47 × 47
conv4_3	3 × 3	1	1	1	1 × 512 × 47 × 47
conv5_1	3 × 3	2	1	2	1 × 512 × 47 × 47
conv5_2	3 × 3	2	1	2	1 × 512 × 47 × 47
conv5_3	3 × 3	2	1	2	1 × 512 × 47 × 47
fc6	7 × 7	12	1	4	1 × 4,096 × 47 × 47
fc7	1 × 1	0	1	1	1 × 4,096 × 47 × 47
fc-final14_p	1 × 1	0	1	1	1 × 15 × 47 × 47
upsample_p	16 × 16	4	8	1	1 × 15 × 376 × 376

Note. DCNN: dilated convolution neural network.

the part segmentations. This zooming-in every object instance and rescaling facilitates easier training and inference for DCNN module2. However, the issue with training parts conditioned to object segmentation is that the error resulted by the object segmentation module propagates to the parts segmentation module. This issue could be easily seen in the segmentation results presented by Dai et al. (2016). Therefore, we feed the cropped image to DCNN module2 instead of filter responses of DCNN module1, while training

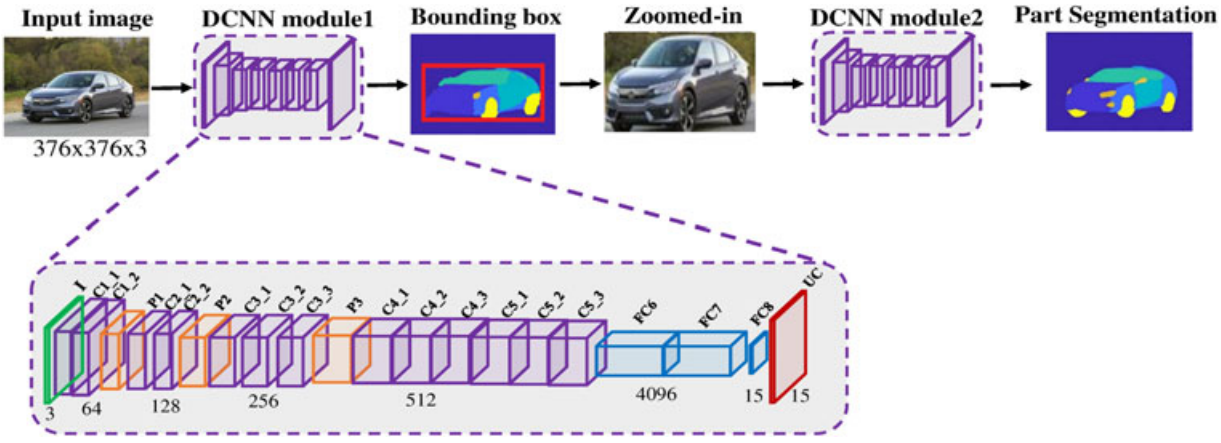

FIGURE 7 The architecture of our cascaded DCNN. The proposed network consists of two DCNN modules stacked together. The part-segmentation output from the first module of the cascade (DCNN module1) is used to compute an overall mask of the vehicle. A tight bounding box extracted from this mask is used to zoom-in the vehicle instance from the input image. This zoomed-in image is then fed to the second module (DCNN module2) of the cascade, which significantly improves the performance of pixelwise segmentation. This procedure of zooming-in to individual instances ensures that DCNN module2 is trained on images of the same scale, making the inference easier and more accurate. Both the modules of the cascade are trained separately so that the error vehicle segmentation from the first module is not propagated to the second module. The detailed architecture of the individual DCNN modules is similar and is presented in Table 1. DCNN: dilated convolution neural network [Color figure can be viewed at wileyonlinelibrary.com]

each layer of DCNN modules is initialized by the weights of the baseline network provided by B. Zhou et al. (2017). We use the pixelwise soft-max loss function given by Equation (5), (6) to train both the networks of the cascade. With the proposed refinements in the architecture, we are able to obtain a significant improvement in the performance for part segmentations. For the task of visual navigation, we are relying on our networks for accurate part localization and segmentation. However, such tasks require the vehicles to be seen from even oblique poses, which are generally not present in training data set. Thus, to enhance the segmentation performance, we further propose two refinements as described in the following subsections.

7.1.2 | Synthetic data augmentation

The PASCAL-part data set consists of images of vehicles taken by persons not explicitly for visual inspection tasks. Thus, there exist numerous viewpoints from which the vehicle has not been previously seen. To cater to the deep network's need for large data set with sufficient viewpoints, we augment the training data with images generated by rendering the 3D models of car from several variations of viewpoints and distance. We have manually annotated ten 3D models of car, by assigning the labels for the desired parts to vertices of computer-aided design (CAD) models. These models are rendered using the Gazebo platform (Koenig & Howard, 2004) from 32 × 6 × 3 viewpoint variations comprising 32 yaw variations, 6 height changes, and 3 discrete levels in depths to generate our data set of 5,760 synthetic images. It is known that data augmentation improves the performance of a network. For example, it was reported by Oliveira et al. (2018) that color and spatial augmentation has significantly improved the performance. We further achieve a noticeable improvement in the accuracy pertaining to synthetic multiview data augmentation. We evaluate the performance upgrade due to the data augmentation in Section 8.4.

7.1.3 | Multiview segmentation fusion

In Section 7.1.2, we use data augmentation to prepare our network to foresee various viewpoints; however, it is exhaustive to cover all the possibilities in SE(3) space. Any error in segmentation could lead to a collision with the vehicle, since our navigation pipeline relies on segmentation for localization. Therefore, we use optical flow-based image warping to transform the part-segmentation label for a given pixel in previous image to its corresponding location in current image. Provided the coordinates of ith pixel x_i^j in previous camera frame $\mathcal{F}_j$, the warped image coordinates

$$x_i^k = \phi\left(x_i^j, \eta\right) = \pi\left(T(\eta)\pi^{-1}\left(x_i^j, Z_j\left(x_i^j\right)\right)\right) \tag{8}$$

are computed by the warping function $\phi(\cdot)$, which transforms pixel x_i^j to the current camera frame $\mathcal{F}_k$ based on the depth $Z_j(x)$ at x_i^j in image I_j and pose η (Ma et al., 2017), where T denotes the camera Euclidean camera transformation matrix and the function $\pi(\cdot)$ refers to the projective transform induced by the camera at the current pose η. However, in our scenario, the depth is not known, and we tackle this issue by assuming that the motion between two iterations is small. Thus, by estimating the optical flow between two subsequent images I_j and I_k, we can forward warp the segmentation mask from I_j to I_k.

We further enhance the segmentation by fusing the warped segmentation of previous view with the current segmentation using maximum a posteriori probability (MAP) estimate thereby producing a multiview consistent output. Ma et al. (2017) used warping-based multiview segmentation association followed by the Bayesian fusion, which improved the segmentation accuracy. For a pixel j, provided the segmentation class labels for the sequence images up-till current frame $(z_1, z_2,\ldots,z_t)$, the predicted label for the current frame at the current pixel j is estimated using the Bayesian fusion:

$$p(j|z_{1:t}) = \frac{p(z_t|j, z_{1:t-1})p(j|z_{1:t-1})}{p(z_t|z_{t-1})}. \tag{9}$$

Assuming the measurements satisfying independent and identically distributed and using log-odd notation, the update rule simplifies to summation of the likelihood term repeated over sequence of frames:

$$\log(p(j|z_{1:t})) = \log(p(z_t|j, z_{1:t-1})) + \log(p(j|z_{1:t-1})) - \log(p(z_t|z_{t-1})). \tag{10}$$

7.1.4 | Data sets

CNNs often require a large amount of data for training, moreover training a segmentation network is even more exhaustive, since every pixel has to be labeled. Thus, in contrary to classification and object detection data sets, there are very few data sets for image segmentation. Due to the increase in categories, part segmentation is highly laborious, as a result there are only two principal data sets with focus on part-level semantic segmentation having a sufficient number of instances (a) Scene Parsing data set (B. Zhou et al., 2017): This data set consists over 15,000 instances of cars and persons each. However, the part-level labels are not accurate as reported by B. Zhou et al. (2017), which introduces unnecessary noise into the system. (b) PASCAL-parts data set: This data set is composed of 10,103 images taken for PASCAL VOC challenge (Everingham et al., 2015) and provides part annotations for each of 20 PASCAL classes. For the task of autonomous inspection, we are interested in cars as category, B. Zhou et al. (2017) is the only paper that reports part-segmentation results on Scene Parsing data set for the car category; however, the ground truth labels for the parts are noisy, which refrains from obtaining a valid benchmark. Another paper that showcases the results for part segmentation is from Oliveira et al. (2018); however, they have trained their network on the person category. Thus, for the comparison purposes, we train our networks on persons as well. Following the data set selection guidelines described by Oliveira et al. (2018), we merged labels at two granularity levels: coarse and fine. For the coarse version, we consider four labels (head, torso, arms, and legs). In the finer version, we have 14 labels discriminating

even between the left and right sides of the person (head, torso, upper right arm, lower right arm, right hand, upper left arm, lower left arm, left hand, upper right leg, lower right leg, right foot, upper left leg, lower left leg, and left foot). Furthermore, similar to Oliveira et al. (2018), we divide the persons data set into training and validation sets. For coarse granularity, we use 70% data for training and 30% for testing the network and for fine granularity, we use 80% data for training and 20% for testing the network as used by Oliveira et al. (2018).

7.1.5 | Network training

We train all the DCNN baseline networks and its refinements and cascades separately. For all the networks, we initialize the weights from the category-level segmentation weights provided by B. Zhou et al. (2017) and fine tune the networks on PASCAL-parts data set. The solver is based on SGD since the batch size selected is 1. The solver's parameters are as follows: Learning rate is initialized with 0.00003 and momentum was constant at 0.9. We use step learning rate policy, and the learning rate is reduced to one-tenth after every 70K iterations. We have trained our networks for 400K iterations, which takes around 48 hr on a Titan-X GPU (Nvidia Corporation, Santa Clara, CA).

7.1.6 | Evaluation metric

For dense pixelwise segmentation, two metrics are reported as a standard practice: (a) pixelwise accuracy and (b) IOU. Let n_{ij} be the number of pixels of class i predicted to belong to class j, where $t_i = \sum_j n_{ij}$ be the total number of pixels of class i. The pixel accuracy is then given by $\text{PA} = \sum_i n_{ii} / \sum_i t_i$. The pixel accuracy takes into account also the prediction of background pixels, and the background pixels cover of the majority of the images. Therefore, pixelwise accuracy is not considered as optimal metric to evaluate the performance of segmentation. Although considering the background pixels, when computing the pixelwise accuracy is not essentially futile, since the background prediction is important to avoid false positives (Oliveira et al., 2018). Therefore, we evaluate our segmentation network on IOU, along with pixelwise accuracy, where IOU is computed as $\text{IOU} = (1/N) \sum_i n_{ii} / (t_i + \sum_j n_{ji} - n_{ii})$.

7.2 | Part-IBVS

PBVS requires accurate estimation of current camera pose in the Cartesian space. However, such knowledge is generally not available or could be susceptible to noise as in our scenario. IBVS in contrast controls the camera motion directly in the image space. This is achieved by considering some geometrical primitives as visual features, such as points, lines, and regions, followed by moving the camera such that the configuration of these features matches a desired configuration. In classical IBVS approaches, keypoints are considered as visual features $\mathbf{s} = [x_1, y_1, \ldots, x_N, y_N,]$. The control law is then defined by assigning the camera a velocity $\mathbf{v}_c$ that minimizes the following objective function:

$$\begin{aligned} \mathcal{L} &= \tfrac{1}{2}(\mathbf{s} - \mathbf{s}^*)^{\mathsf{T}}(\mathbf{s} - \mathbf{s}^*) \\ \Rightarrow \mathbf{v}_c &= -\lambda \nabla \mathcal{L} \\ \Rightarrow \mathbf{v}_c &= -\lambda \mathbf{L}_{\mathbf{s}}^{+}(\mathbf{s} - \mathbf{s}^*), \end{aligned} \tag{11}$$

where $\mathbf{L}_{\mathbf{s}}$ is the interaction matrix that maps feature velocity in the image space to the camera velocity $\mathbf{v}_c$ in the Cartesian space. For a 3D point $[X, Y, Z]$ as visual feature, the interaction matrix is given as (Chaumette & Hutchinson, 2006)

$$\mathbf{L}_{\mathbf{s}} = \mathbf{L}_{\mathbf{x}} = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}. \tag{12}$$

In this paper, the objective for Part-IBVS is to attain a desired pose with respect to a given part. However, the part belongs to a novel instance of a car, which has not been previously seen, that is, the instance in the desired image is different from the current instance. Thus, classical IBVS could not be directly used for achieving our task. Therefore, in this paper, we use the segmentation masks provided by our part-segmentation CNN for visual servoing as shown in Figure 8. The objective is modified as to place the camera attached to the MAV such that the center of the segmentation coincides with the camera center and the area of the segmentation equals a predetermined area.

To accomplish this modified task, we consider image moments as visual features (Chaumette, 2004). Since our MAV is underactuated and only four degrees of freedom (DOF) could be controlled. Therefore, we employ the following image moments as visual features: (i) centroid of the segmentation and (ii) the area occupied the segmentation in the given image.

$$\begin{aligned} \mathbf{s} &= [x_g \quad y_g \quad A], \\ \mathbf{s}^* &= [0 \quad 0 \quad A^*]. \end{aligned}$$

The interaction matrix for these visual features is given by

$$\mathbf{L}_{\mathbf{s}} = \mathbf{L}_{\mathbf{x}} = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \\ 0 & 0 & \frac{-1}{Z} & 0 & 0 & 0 \end{bmatrix}. \tag{13}$$

Finally, to control the MAV, we use the exponential decay controller used by classical IBVS approaches presented in Equation (11).

7.3 | Reverse-PBVS

The objective of this module is to navigate the MAV back to the desired pose of the current side $\mathcal{F}_s^*$ from current pose resulted from Part-IBVS module $\mathcal{F}_i^*$. Since both $\mathcal{F}_s^*$ and $\mathcal{F}_i^*$ are in the same frame $\mathcal{F}_X$, therefore, classical PBVS could be directly applied to achieve this objective. Similar to the PBVS module, we rely on the noisy odometry provided by the MAV for state estimation. Thus, the state estimation could be incorrect and an IBVS refinement could be applied for rectifying the state estimation error. However, IBVS refinement will consume additional time for alignment, we, therefore, skip the IBVS

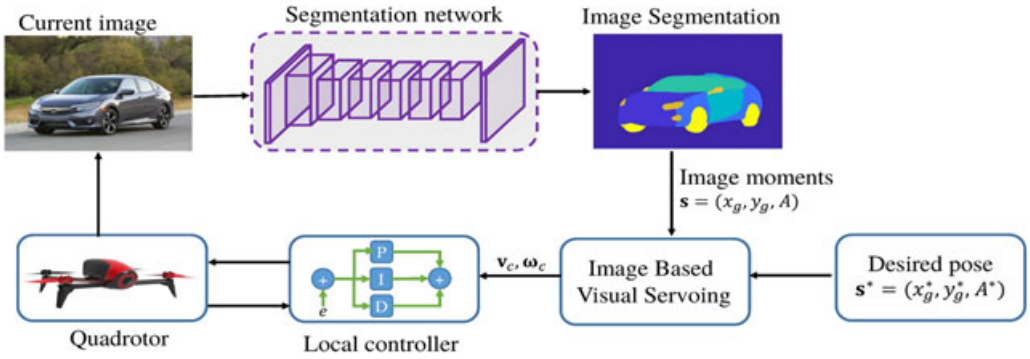

FIGURE 8 Part-servoing pipeline: We propose dilation-based convolutional neural network for extracting pixelwise part segmentation. Image moments (centroid and area) are extracted from these segmentation masks as visual feature for IBVS algorithm. The velocity control commands issued by IBVS controller are smoothly tracked by MAV's local controller, and finally, the loop is closed by MAV's vision sensor. IBVS: Image-Based Visual Servoing; MAV: Micro Aerial Vehicle [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 2 Overall performance of keypoint network on the PASCAL 3D data set with 14 body parts

Approach	PCK (%; $\alpha = 0.1$)
Tulsiani and Malik (2015)	81.3
Li et al. (2017)	81.8
Hourglass architecture (Newell et al., 2016)	**93.4**

Note. Best value is presented in bold. PCK: Percentage of Correct Metrics.

refinement for this reverse-PBVS module, since we observed only small pose errors while inspecting three-to-four parts. When we switch sides, the side navigation again employs IBVS refinement.

8 | EXPERIMENTS AND RESULTS

We evaluate our approach extensively through a series of experiments. Initially, we assess the performances of individual components of the pipeline on both qualitative and quantitative measures. Section 8.1 presents the results of the keypoint network from our previous work (Kumar et al., 2017; Murthy et al., 2017) and in Section 8.2, we showcase the results of our segmentation network. Sections 8.3 and 8.4 are dedicated to the refinements proposed in this paper to improve the part-segmentation performance. Finally, in Section 8.5, we implement our approach on a Parrot Bebop-2 MAV and perform multiple field experiments for our completely autonomous inspection pipeline.

8.1 | Evaluating keypoint network

Herein, we evaluate the accuracy of our 2D keypoint localization framework. We trained and evaluate our network on annotated images of cars from publicly available PASCAL 3D data set (Xiang et al., 2014) for cars category. All the images while training and testing were annotated for 14 part-based keypoints. To evaluate the performance of our network based on hourglass architecture

FIGURE 9 Qualitative results showing the 2D keypoint localization performance of the used architecture. Top seven keypoints per instance are shown (in accordance with the confidence scores output by the CNN). Discriminative features are extracted consistently across instances, pose variations, and occlusions. The last row shows some failure cases. CNN: convolutional neural network [Color figure can be viewed at wileyonlinelibrary.com]

proposed by Newell et al. (2016), we use the standard Percentage of Correct Metrics (PCK) used by Tulsiani and Malik (2015). In our analysis, we use a very tight threshold of 2 pixels to determine whether or not our keypoint estimate is correct. We further compare the accuracy obtained by our approach for the car category with the state-of-the-art part-based keypoint prediction approaches (Li et al., 2017; Tulsiani & Malik, 2015). Table 2 shows the keypoint localization accuracy obtained by the hourglass network architecture. The results indicate a significant performance boost in the task of keypoint localization, which also improves the performance of the pose-induction module. A few keypoint predictions are shown in Figure 9. It can be seen that our approach robust to even large appearance variations. The figure also shows some failure cases, which are due to the change in shape, such as an opening of trunk or parts not visible.

8.2 | Evaluating part-segmentation network

The objective here is to thoroughly validate the segmentation network, since semantic segmentation of parts is the central component of our visual inspection pipeline. A wrong segmentation could lead the MAV in an incorrect direction due to lack of any manual supervision during the test time. Furthermore, our keypoint network requires bounding box of the car as additional prior for keypoints predictions. This bounding box is computed by the first cascade of our network. Another option while computing the bounding box is YOLO (Redmon, Divvala, Girshick, & Farhadi, 2016), faster-RCNN (Ren, He, Girshick, & Sun, 2015), which again requires an additional network that means more computation during test time. Also, YOLO and faster-RCNN are not trained for zoomed-in part predictions, as result of any zoomed-in car they result in highly inaccurate bounding box predictions. These situations arise frequently in our scenario of part inspection, which makes YOLO and faster-RCNN not well suitable for this task.

To compare our results with Oliveira et al. (2018), we test our network for both coarse and fine part predictions. Similar to Oliveira et al. (2018), for the coarse parts data set, we retain four labels (head, torso, arms, and legs) and for fine parts, we predict all the 14 parts of person category from PASCAL-parts data set. Oliveira et al. (2018) have trained their network on 80% and 70% images, respectively, for coarse and fine part predictions. However, Oliveira et al. (2018) do not provide an extensive list of training and testing images; therefore, we randomly select 70% images for training and the remaining 30% images for testing the coarse predictions. Similarly, the 80:20 ratio was used for training and testing the segmentation network on fine parts.

Figure 10 shows the predictions by our network for a few images from PASCAL-parts data set. It can be seen that even for multiple humans and at different scales, our network is able to successfully capture the part information. We compare our network with existing networks on both fine and coarse label predictions. It can be seen from Table 3 that our baseline network, which is a single DCNN module and is trained without any data augmentation, outperforms Part-Net for segmentation of almost every part giving us significant improvement in the mean segmentation. Table 4 compares the performance on coarse data set with four parts. Our network results are at par with M-Net (Heavy) and significantly better over other networks for IOU.

Furthermore, our network achieves over 5% improvement over state of the art in pixelwise accuracy, which indicates that the false positives are less since the pixelwise accuracy also includes the

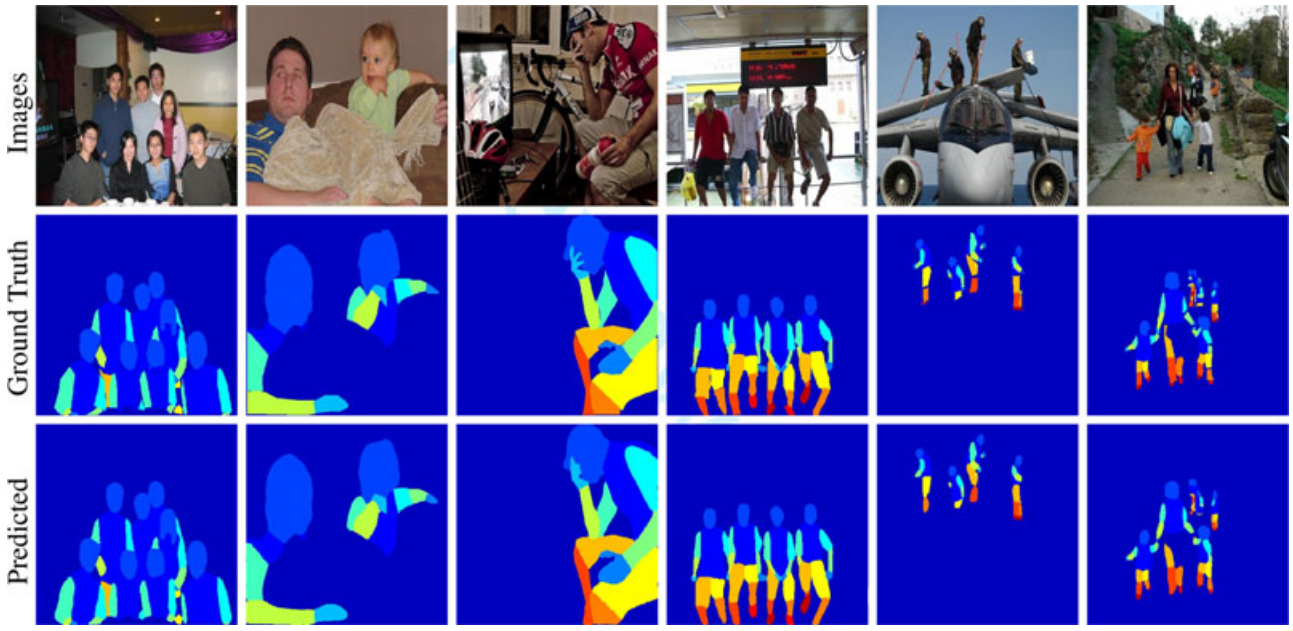

FIGURE 10 Qualitative results for pixelwise segmentation of 14 parts on person category from PASCAL data set. Here, we compare the performance of the baseline DCNN network with single DCNN module, which significantly improves the segmentation performance. DCNN: dilated convolution neural network [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 3 Partwise comparison results of our baseline dilated network with existing approaches for segmentation on the PASCAL data set with 14 body parts

Method	Head	Torso	L U arm	L LW arm	L hand	R U arm	R LW arm	R hand	R U leg	R LW leg	R foot	L U leg	L LW leg	L foot	Mean of all parts
FCN	74.0	66.2	56.6	46.0	34.1	58.9	44.1	31.0	49.3	44.5	40.8	48.5	47.6	41.2	48.8
Part-Net (spatial)	81.8	78.0	69.5	63.1	59.0	71.2	63.0	58.7	65.4	60.6	52.0	67.9	60.3	50.0	64.3
Part-Net (spatial + color)	84.0	81.5	74.1	68.0	64.0	75.4	67.4	61.9	**72.4**	67.1	56.9	**73.0**	66.1	57.7	69.2
Ours (dilated net baseline)	**92.9**	**84.1**	**76.2**	**69.9**	**67.0**	**77.2**	**71.0**	**68.6**	72.2	**70.1**	**67.0**	71.3	**68.9**	**72.2**	**73.5**

Note. We have evaluated our network on human category to benchmark with Oliveira et al. (2018) Best value is presented in bold. L: Left; LW: Lower; R: right; U: upper.
Best value is presented in bold.

background labels. Moreover, the average time taken for forward pass by our network on a Titan-X is 161.66 ms, which is near real time and is more suitable for autonomous navigation-based applications that require online processing.

8.3 | Improvements due to two-stage cascading

We now evaluate the cascaded architecture presented in this paper on PASCAL-parts data set for car category. Similar to Section 8.2, we divide the data set in 80:20, and use 80% data for training and rest for testing. We compare the performance of sequential cascading with the baseline DCNN for parts segmentation. Figure 11 shows the predictions by both the networks on PASCAL parts, and the quantitative evaluation on the same is presented in Table 5. It can be seen from Table 5 that cascading improves the segmentation performance significantly both in terms of pixel accuracy and IOU. The reason accounting for this improvement is that segmentation is degraded on cars that are at a farther distance as could be seen in Figure 11. Furthermore, the output from the first module of the cascade is normalized for scale for each car which, when passed to the second module, helps in capturing fine details of the parts, thus improving the segmentation performance for individual parts significantly on the cars that are farther, and a marginal improvement even on the cars that are closer due to uniformity in the scale. Note that only B. Zhou et al. (2017) presents the part-segmentation results on car category on ADE20K data set. However, annotations provided by ADE20K are noisy as reported by B. Zhou et al. (2017); therefore, we do not directly compare with B. Zhou et al. (2017).

8.4 | Effect of proposed refinements

In this section, we evaluate the effect on segmentation performance due to the refinements proposed in this paper. Since these proposed refinements in part-segmentation networks are based on synthetic data augmentation and multiview Bayesian fusion, which are not available on public data sets, therefore, we report results only in simulation, where the ground truth could be easily computed.

8.4.1 | Synthetic data augmentation

Vehicle part inspection requires viewing cars from oblique angles, which are generally not available in images provided by public data sets. We, therefore, create data set of ten 3D CAD models and manually annotate the meshes. These models are rendered using the Gazebo platform from 32 × 6 × 3 viewpoint variations comprising 32 yaw variations, 6 height changes, and 3 discrete levels in depths to generate our data set of 5,760 synthetic images. These data are augmented to the PASCAL-parts data set for training the network. We have seen a noticeable improvement in the network performance,

TABLE 4 Overall performance on segmentation on the PASCAL data set with four body parts

Method	IOU	PA	Precision	Recall	Time (ms)
FCN (Long et al., 2015)	57.35	71.79	77.28	67.92	150
SegNet (Badrinarayanan et al., 2015)	45.22	44.82	49.88	80.71	**47.7**
ParseNet (Liu et al., 2015)	64.25	70.02	74.66	78.95	88
Part-Net (Oliveira et al., 2018)	78.23	85.47	86.00	87.78	225
Fast-Net (Oliveira et al., 2018)	81.92	88.81	88.74	90.04	48.7
M-Net (Oliveira et al., 2018)	78.15	84.95	86.29	87.60	130
M-Net (heavy)	**84.62**	91.51	91.47	**90.57**	345
Ours (Dilated Net baseline)	82.98	**96.72**	**91.61**	89.06	161.7

Note. Our network performs at par with state of the art in IOU while giving a significant boost in pixelwise accuracy (PA).
Best value is presented in bold. IOU: intersection over union.

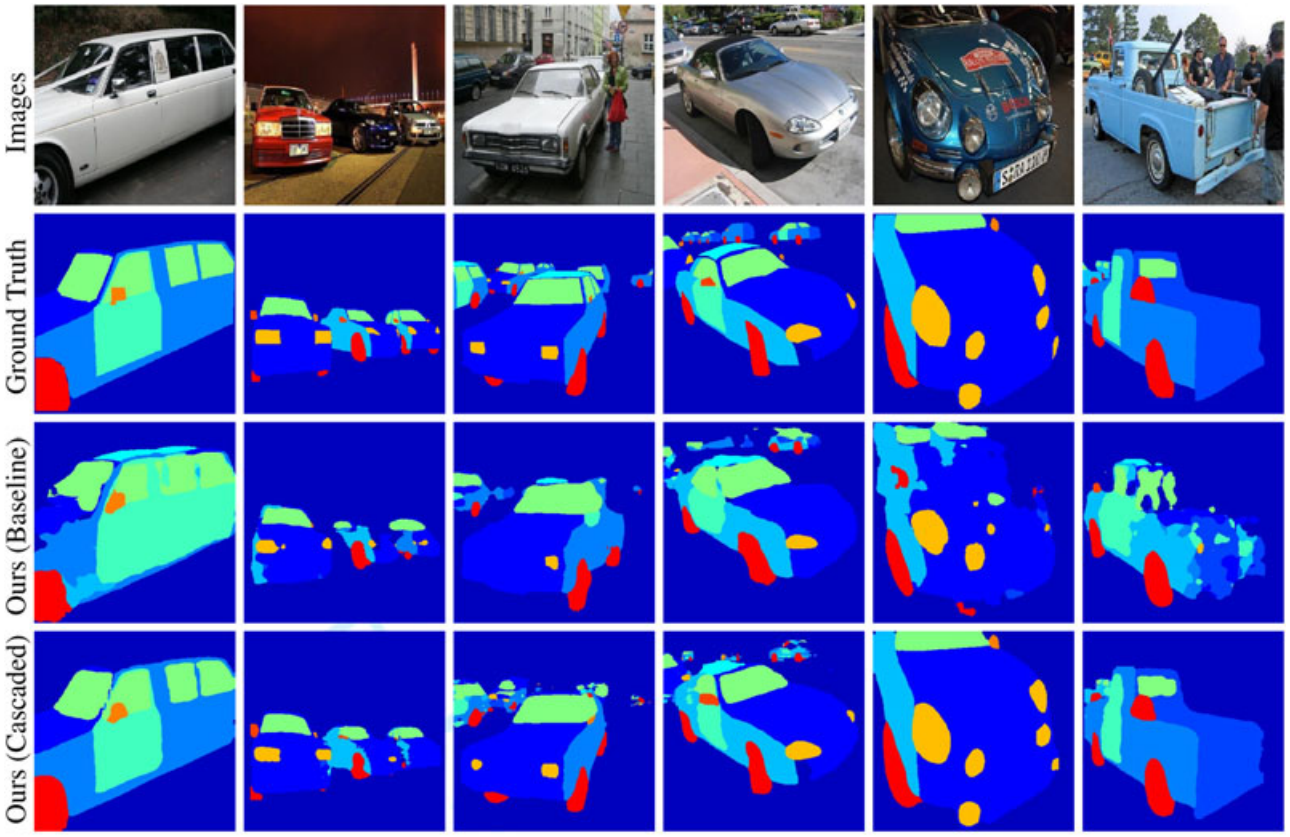

FIGURE 11 Qualitative results for pixelwise segmentation of 11 parts on cars category from PASCAL data set. Here, we show the advantage of cascaded architecture over the baseline, which is a single DCNN module. DCNN: dilated convolution neural network [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 5 Results for part segmentation on the cars category of PASCAL data set using our baseline dilated network and the proposed cascaded architecture

Method	Front	Back	Left	Right	Roof	Door	Window	Headlight	L-Mirror	R-Mirror	Wheel	Mean IOU	PA
Dilated Net (baseline)	50.2	23.5	24.1	30.4	17.8	32.7	58.0	34.6	14.9	7.8	58.2	32.02	49.01
Dilated Net (cascaded)	86.0	78.0	75.3	68.5	70.5	73.8	80.9	74.9	48.0	56.1	83.9	61.23	86.20

Note. IOU: intersection over union; PA: pixelwise accuracy.

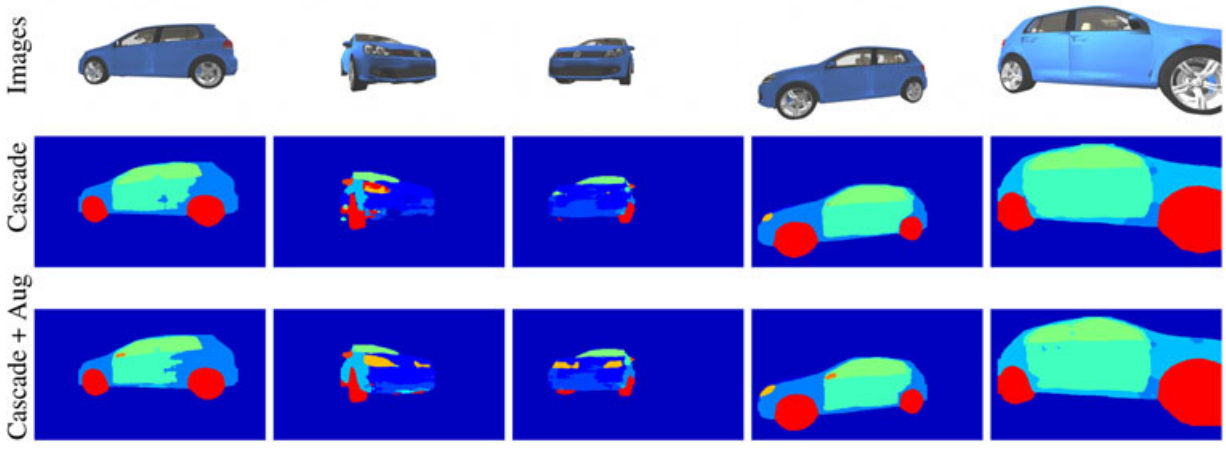

FIGURE 12 Qualitative results for improvements due to data augmentation refinement over our cascaded DCNN on the Gazebo simulation for different viewpoints. Note the improvement in the segmentation of mirror and headlights. DCNN: dilated convolution neural network [Color figure can be viewed at wileyonlinelibrary.com]

especially for oblique views. Figure 12 qualitatively compares the performance of our cascaded network trained only on PASCAL-parts data set with that trained over the multiview augmented data set. Although only the data from the simulation are augmented, still we obtain substantial improvements even in real images of vehicles while executing the complete visual inspection task.

8.4.2 | Multiview fusion

Our pipeline relies on part segmentation, for which the network has been extensively trained in both simulation and real data. However, we employ IBVS for navigation, which could result in camera viewpoints in the SE(3) space, in which the network may be unable to generalize. Therefore, we use warping-based multiview fusion approach discussed in Section 7.1.3 for improving the segmentation. The comparative results are shown on the simulation environment only, since there is no public data set with such viewpoint variations for part segmentation. The setup comprises an MAV platform, for which we use the RotorS Gazebo (Furrer, Burri, Achtelik, & Siegwart, 2016) library, and twelve 3D CAD models of cars. We use one car among them as the template model Y and the rest are used as previously unseen novel instances X for the inspection. For every car, the MAV is initialized at a random pose with no prior knowledge of the environment. The proposed pipeline is further used by the MAV for servoing to all the required parts (eight parts per car) in a predefined order. We showcase the improvement in inspection performance by integrating the Bayesian fusion with the visual inspection pipeline, as depicted in Table 6. It can be seen from the table that the Bayesian fusion not only increases the convergence rate but also decreases the residual error. We encourage the readers to refer the supporting information[3] for details regarding the simulation experiments, such as experimental setup, qualitative results, and additional quantitative results, for the complete visual inspection pipeline.

8.5 | Evaluating the autonomous inspection pipeline on field experiments using MAV

Finally, as a final proof of concept and viability of our approach, we validate our pipeline for autonomous inspection of vehicles using the low-cost Parrot Bebop-2 MAV. In our experiments, we showcase our results of 16 runs in the field on seven different car models (Suzuki Alto, Tata Indigo, Hyundai Accent, Mahendra E2O, Hundai Creta, Mahendra Scorpio, and Suzuki Omni van) with an autonomous flight time of over 190 min. We next describe the setup used and outline the scenarios, in which we conducted the experimentation.

8.5.1 | Experimental setup

The experimental setup shown in Figure 13a comprises a Parrot Bebop-2 MAV, a vehicle from Alto, Indigo, Accent, E2O, Van, Scorpio, and Creta and an ROG GL552VX laptop (AsusTek Computer Inc.). The sensory suite of a Parrot Bebop MAV consists of a front-facing monocular RGB camera, Global Positioning System (GPS), IMU, and Wi-Fi, as illustrated in Figure 13b. We employ the front-facing camera for capturing the images, although the front camera consists of a fish-eye lens, we use the undistorted images so that the pin-hole camera model could be applied for visual servoing. Bebop-2 MAV provides odometry by fusing IMU measurements with its optical flow sensor. However, this odometry is noisy and tends to drift over time. In our experiments, we use this odometry for PBVS and inverse-PBVS; however, as mentioned previously, we do not require accurate odometry since we employ IBVS refinements and Part-IBVS for better localization. Since MAVs are underactuated, only four DOF control tasks were selected for visual servoing. We use ROG GL552VX laptop for processing the images captured by the MAV and generating control commands. The image captured by the MAV, and corresponding control commands generated by the visual servoing controller are exchanged between the system and MAV over Wi-Fi. ROG GL552VX laptop features a Core i7 CPU, Nvidia GTX 960M GPU and 16 GB RAM. The CNN forward pass for keypoint prediction as well as part detection was performed using this laptop computer, and it takes 800 ms for one iteration to complete on the laptop, which includes the forward pass of the cascade network, keypoint network, and optical flow computation. Note that for inspection tasks the MAV moves to three different sides of the vehicle requiring to fly through large camera transformations between nonoverlapping scene with a total autonomous flight time of around 12 min for every flight, without any manual intervention during the flight. Also, note that we report the quantitative results using an approximate odometry provided by the MAV as in the real world, it is difficult to accurately predict the position of an MAV. In every experimentation scenario, we selected one vehicle from the four models and the MAV was randomly placed around the car such that the car was in the field-of-view of the MAV. The car was parked outdoors, and the navigable area for the MAV (5 m × 5 m) was obstacle-free; however, the car itself was an obstacle and our approach ensured no collision.

TABLE 6 Quantitative results showing the improvement in performance of visual inspection attributed to the Bayesian fusion

Method	Mean residual error	Convergence rate (%)
Without Bayesian fusion	0.4250	78.21
With Bayesian fusion	0.2625	88.54

Note. CAD: computer-aided design; MAV: Micro Aerial Vehicle.
Note. We validate the effect of the Bayesian fusion module in the Gazebo simulation setup, where the MAV starts from a random pose and servoes to the required parts of a novel car. In total, 12 car instances (CAD models) were used for the experiment, and the MAV was required to servo eight parts per car (i.e., 96 parts for all cars). The Convergence rate depicts the number of parts, and MAV was able to visit correctly out of 96 parts, whereas the residual error, measured in percentage of total image size, describes the mean error between the area of the part after servoing and desired area of that part. It can be seen that the Bayesian fusion significantly improves the inspection performance.

[3]Supporting information: https://robotics.iiit.ac.in/people/harit.pandya/vi_inspection/vi_supp.eps

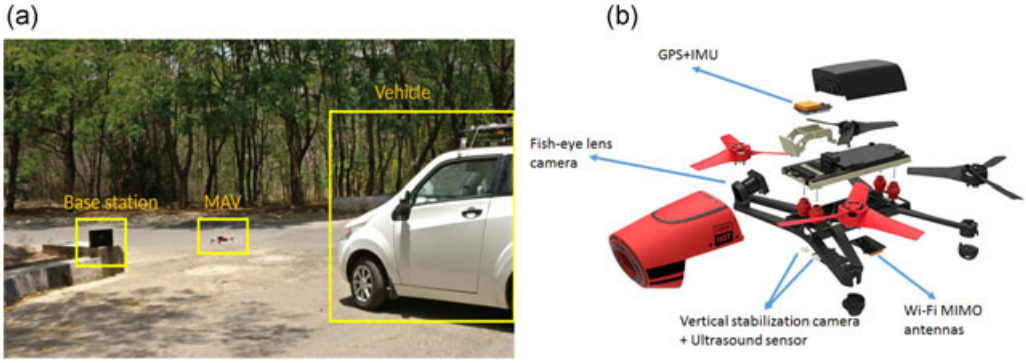

FIGURE 13 Experimental setup. (a) The setup used for the field experiments. It consists of a vehicle parked outdoors, an MAV, which starts at a random pose with respect to the vehicle and a base station for processing the images captured by the MAV and issuing control commands to the MAV. (b) For our experiments, we have used Parrot Bebop-2, which is a low-cost off-the-shelf MAV. Its sensory suite includes a front-facing camera, Wi-Fi antennas, GPS, IMU, downward facing camera for stabilization, and an ultrasonic sensor. GPS: Global Positioning System; IMU: inertial measurement unit; MAV: Micro Aerial Vehicle [Color figure can be viewed at wileyonlinelibrary.com]

8.5.2 | Evaluating visual inspection

After exhaustive testing of individual components, we conduct experiments for 16 experiments on seven different vehicles (Suzuki Alto, Hyundai Accent, Tata Indigo, Mahindra E2O, Hundai Creta, Mahendra Scorpio, and Hundai creta). These cars are selected to cover the four major sub-categories of vehicles (sedan, hatchback, MUV, SUV, and Van). In every experiment, we place the MAV at a random initial pose such that car is visible. The MAV starts with no prior knowledge about the environment, it then employs the pose-induction pipeline to estimate its pose with respect to the given vehicle and servoes to the eight essential parts (4 wheels, 2 headlights, and 2 mirrors) for every car. We do not test on taillights because taillights are not part of PASCAL-part data set. With sufficient training sample, our approach could easily be extended to other parts or object categories. Here, we assume that the workspace is obstacle free and car is visible in the initial random pose.

The qualitative results are presented in Figure 14. These results show that the image captured by the MAV from the initial pose, from the resultant pose attained after side-navigation step and the final image of every part captured after the part navigation step along with their segmentation masks computed by our network. It could be seen that with our approach. The MAV is able to visit the correct parts despite different shapes of essential parts and their configurations. Additional qualitative results are shown in Figure A1 in the Appendix.

The objective of the visual inspection pipeline is to navigate the MAV so that it visits all the required parts and capture their images. Thus, for quantitatively evaluating our approach we compute the area of every part captured from the resulting pose of Part-IBVS and compare it to the desired area of that part. These areas are selected as 1% of the total image area for mirrors, 2% of the total image area for the headlights, and 10% of the total image area for other parts. The desired area can be selected as per requirement. However, it should be noted that for achieving a larger desired area, the MAV has to fly closer to the vehicle, which could lead to collisions. The quantitative results are presented in Table 7. Out of total 128 parts to be inspected our approach failed to converge for 15 parts, which are marked as '–' in the table. Thus, it can be seen that our approach is able to converge for more than 88% of the times and the average error in the area is 0.75%, which is under acceptable limits. A few failure cases are also shown in Figure 14, which occur due to incorrect pose induction and wrong segmentation produced by the network. Note that while reporting the quantitative results presented in Table 7, the reported area of the part (A) in the final image is computed by manually annotating the part rather than relying on the segmentation given by the network. The motivation behind this is to obtain the ground truth area of the part as the segmentation mask provided by the network could be incorrect.

8.5.3 | Evaluating visual servoing

Once we have evaluated our approach on visual inspection parameters, we now show the performance of our approach on visual servoing metrics, specifically on error in visual features (image area and image centroid), camera poses error, velocity profile, and camera trajectory for PBVS, reverse-PBVS, and Part-IBVS. We consider a single test case and plot evolution of these parameters over the entire run. The images captured by the MAV for this run are presented in Figure 14 column 1. PBVS is used to navigate the MAV to the three sides. We, therefore, show the position error (translation in X, Y, Z and yaw) of the MAV for the three sides in Figure 15a. Since the MAV starts from the left side, it quickly attains the desired pose of the left. For the front and the right side, the position error exponentially decreases initially followed by a small increase before converging to the desired pose. This overshooting behavior is due to the higher gains of PBVS controller of the MAV and could also be seen in the MAV's trajectory shown by Figure 15f. In Figure 15b, we show the position error for reverse-PBVS, which navigates the MAV from a part to the corresponding side. Similar to the PBVS, the position error decreases exponentially to the desired pose. For analyzing the Part-IBVS approach, we plot the feature error and position error. The visual features are composed of centroid's

FIGURE 14 Qualitative results for complete autonomous part inspection pipeline with the Bayesian fusion in the outdoor experiment. This figure showcases the results of the entire pipeline for five cars. The first row shows the images captured from a random initial pose. The images captured by the MAV from $\mathcal{F}_s^*$, for all sides are shown in the second, third, and fourth rows. The subsequent rows show the images captured for every part and the corresponding part-segmentation masks predicted by our network. Note that despite the different shapes of cars and starting MAV at random poses, our approach aligns the MAV for visual inspection. The figure also shows some failure cases of our approach where the parts are out of MAV's field-of-view or segmentation mask is not correct. We evaluated our approach in challenging illumination environments, such as shadows and heavy sunlight [Color figure can be viewed at wileyonlinelibrary.com]

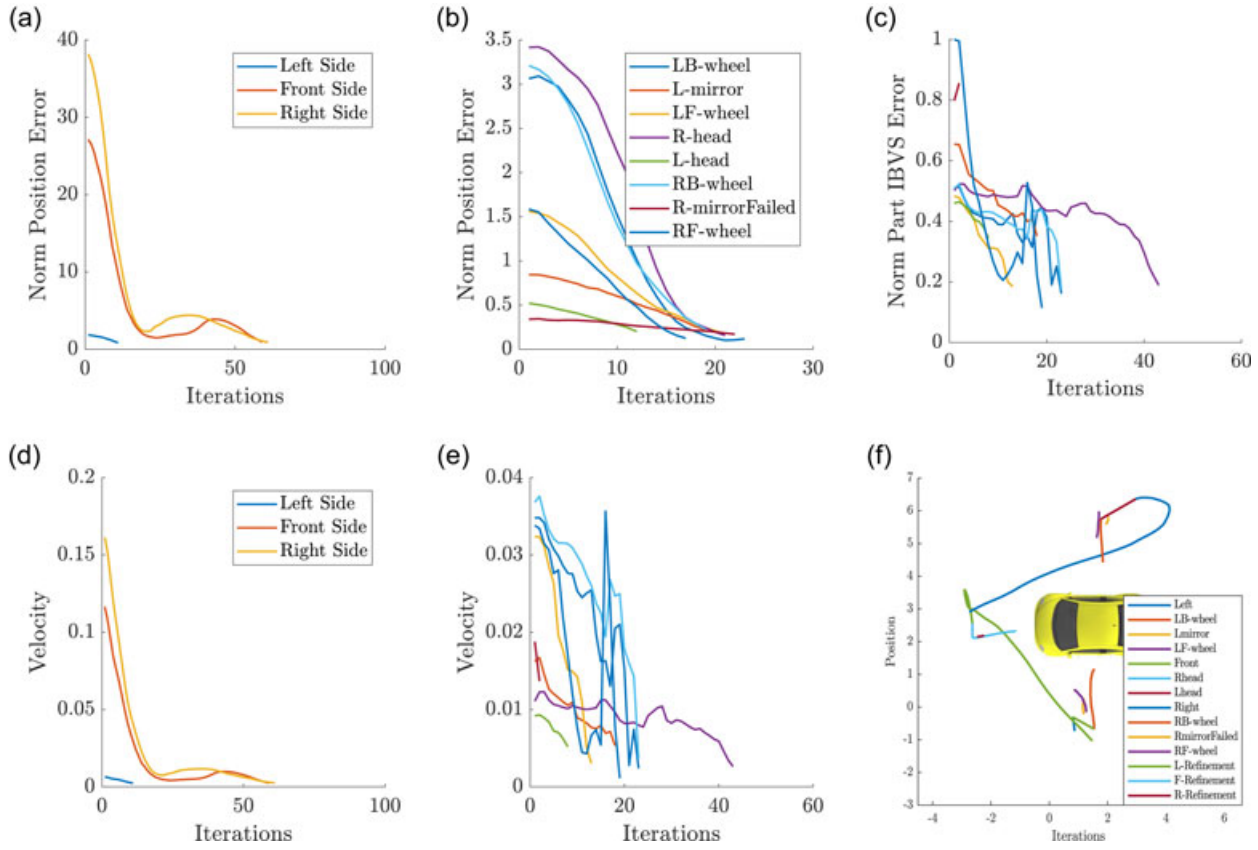

FIGURE 15 Performance of visual servoing for an outdoor run for inspection of the actual car using a Bebop MAV. The performance of both PBVS and Part-IBVS on visual servoing measures are presented here. (a) The error in camera pose ($\|\mathcal{F}_c - \mathcal{F}_s^*\|$) representing translation and yaw for the PBVS. (b) The camera transformation error ($\|\mathcal{F}_c - \mathcal{F}_i^*\|$) for reverse-PBVS to every part. (c) Presents the error in visual features (image moments for the part-segmentation mask) over time for every part using Part-IBVS. The norm error between the current and the desired area of part segmentation abs$(A - A^*)$ in the percentage of the area of image $(M \times N)$ and centroids current and desired segmentation mask in the pixel $\|[x_g - x_g^*, y_g - y_g^*, A - A^*]\|$ is plotted. (d, e) The norm of camera velocities for PBVS and Part-IBVS, respectively. (f) The camera trajectory for inspection to every part combining our hierarchical PBVS and IBVS approach. The improvement due to IBVS refinement could be clearly seen in (f). The legend for figures (c, e) is the same as (b) and is not shown for clarity. Note that the visual feature errors are not very smooth. This is due to the optical flow errors, which result in incorrect warping when exposed to harsh illumination conditions; however, they exponentially decay to their desired values. Also, note that the trajectory is plotted using the visual odometry provided by the MAV, which is highly inaccurate. Despite such noisy odometry, our approach is able to accomplish the task of visual inspection with complete autonomy. F: front; IBVS: Image-Based Visual Servoing; L: left; LB: left-back; LF: left-front; MAV: Micro Aerial Vehicle; PBVS: Position-Based Visual Servoing R: right; RB: right-back; RF: right-front. [Color figure can be viewed at wileyonlinelibrary.com]

coordinates and area of the part segmentation; therefore, both of them are jointly reduced while performing Part-IBVS. It can be seen from Figure 15c that the error in visual features decreases exponentially. However, it is not strictly decreasing. There are two reasons for this observation: (i) Since both of the visual features are jointly reduced, it may happen that the error in one feature increases, whereas the other feature is optimized in this iterative procedure of Part-IBVS. (ii) Segmentation mask is computed using deep learning, which views every image as a separate sample, thus despite the Bayesian fusion, there are variations in segmentation, which affects the servoing performance. However, the overall error decreases exponentially and the servoing converges to the desired pose. For this vehicle, MAV was unable to attain the desired pose for the right mirror, as segmentation was not accurate; therefore, the visual feature error is unable to converge.

The velocity profile is smooth for the PBVS (refer Figure 15d) and reverse-PBVS steps, whereas it is jittery for the Part-IBVS step (refer Figure 15e) due to the forward pass delay resulted by our networks. The trajectory visualized from bird-eye view highlights the motion of the MAV toward all the parts and to-and-fro vantage point. Note that the trajectory plotted in Figure 15f is using the inaccurate odometry of the MAV, which is the reason of discontinuity in the trajectory. This also highlights the requirement for IBVS-refinement step. Also, the oscillations due to incorrect gains could be seen as the MAV overshoots and returns to the desired pose. From the results presented, it could be seen that our approach is able to attain the correct areas for all the parts despite such inaccurate odometry and illumination variations. More experimental trials and visualization can be found in the supplementary video in the online version of the paper (refer to the Appendix).

TABLE 7 Quantitative results for complete autonomous part inspection pipeline with the Bayesian fusion in outdoor experiments

Car name	RB-wheel	RF-wheel	R-mirror	R-head	L-head	L-mirror	LF-wheel	LB-wheel
Final area (A) of parts achieved with Bayesian on real cars								
Accent 1	10.6	7.6	0.8	3.9	1.4	0.4	10.8	12.0
Accent 2	9.5	7.8	–	–	1.9	0.5	6.1	8.7
Indigo 1	6.9	12.5	0.9	–	4.6	2.0	9.6	10.0
Indigo 2	–	7.9	4.6	1.0	3.3	1.6	6.7	9.8
Alto 1	11.0	7.7	0.4	2.4	1.4	1.3	7.9	8.3
Alto 2	10.3	6.4	1.3	2.7	1.1	1.3	13.8	11.5
Alto 3	11.8	6.8	0.8	2.5	1.4	–	7.5	12.4
E2O 1	9.8	11.8	0.4	1.5	4.3	0.4	7.3	7.8
E2O 2	6.2	18.6	1.2	2.7	1.5	1.0	6.4	5.8
E2O 3	12.0	12.0	0.6	3.4	1.9	–	6.1	8.7
Creta 1	7.1	4.0	–	1.5	1.6	1.0	8.9	12.8
Creta 2	6.6	3.1	–	2.2	1.4	1.2	6.8	6.0
Scorpio 1	11.9	8.0	–	2.9	2.5	–	5.4	9.1
Scorpio 2	6.7	10.8	–	1.1	6.9	–	6.3	6.2
Van 1	9.9	9.4	–	5.0	2.5	1.6	10.0	7.0
Van 2	7.6	12.3	–	4.0	1.9	–	10.2	6.5
Mean area	9.2	9.2	1.2	2.6	2.5	1.1	8.1	8.9
Desired area (A^*)	10	10	1	2	2	1	10	10

Note. Here, we evaluate our visual inspection pipeline on outdoor experiments for 16 runs on seven cars by measuring the final area (A) of a part captured by MAV in percent of the image area as compared with the desired area (A^*) for that part. The A^* (10% for wheels, 1% for lights, and 2% for mirrors) is a tuning parameter that decides zoom level for images. It can be seen that by using the Bayesian fusion both the failure cases reduce as well as the resultant segmentation error abs($A - A^*$) decrease, that is, the final area is closer to the desired area. The failure cases where the MAV is not correctly aligned with the vehicle are reported as '–.' The areas reported in the above table are computed using the ground truth labels of the corresponding parts in the resultant image. L: left; LB: left-back; LF: left-front; R: right; RB: right-back; RF: right-front.

9 | DISCUSSIONS AND CONCLUSIONS

9.1 | Lessons learned

It is a known issue with deep learning approaches that for novel data, incorrect predictions are performed with high confidence. Therefore, even though the deep networks showcase superior results on public data sets, employing them for visual servoing or navigation could result in a collision. In this paper, we proposed refinements that reinforce our deep networks through multiview data augmentation and the Bayesian fusion to tackle this challenge. Selecting the step size is crucial for visual serving approaches as smaller step size increases the time required for servoing, whereas larger step size could lead to oscillatory behavior. In the field experiments, selecting proper step size is even more important as, on the one hand, it is required to minimize the servoing time as the MAV has limited battery resources and on the other hand overshooting the trajectory due to larger step size could result in a collision with the vehicle. We empirically select a step size that minimizes the time taken for servoing and prevents overshooting the trajectory by MAV.

The time duration for performing a Part-IBVS iteration is 800 ms, this results in a jittery velocity profile and trajectory. The potential solution to this issue includes employing better hardware for the base station or compressing the network so that the speed increases without affecting the performance. As an example, we compare benchmark the performance of our segmentation network on a few graphics processing unit (GPU) models and show the results in Table 8. It can be seen that by using a better GPU, such as Titan-x, can easily give four times boost to the segmentation speed.

9.2 | Failure cases

Our failure cases belong to two categories: First alignment errors, which occur when the deformation in shape of the given instance is large as compared with the template instance. In such cases, not only there are chances of MAV going out of the filed of view of the vehicle but also the overall error in the resultant area of part increases. One

TABLE 8 Processing time for our part-segmentation network on different GPUs

GPU model (Nvidia)	Quadro K2200	GTX Titan-x	GTX 960M	GTX 1080Ti
Processing time (s)	0.5580	0.1354	0.5145	0.1771

Note. It can be seen that using a high-end GPU significantly improves the processing speed.
Note. GPU: graphics processing unit.

potential solution to this problem proposed in our previous work (Pandya et al., 2015) used a linear combination of 3D models of that category. Second, the majority of our failure cases occur when the part-segmentation module is unable to correctly segment the given part. It can be observed from Table 7 that out of 15 failure cases 11 alone are due to incorrect segmentation of mirror by our network. We observed that the segmentation network faces challenges while discriminating small parts with color similar to the vehicle's body.

9.3 | Conclusion

In this study, we have introduced a novel pose-induction framework for visual servoing across instances of an object category. We have used this approach to build a novel framework that systematically embeds semantics into classical controllers. Our approach is able to achieve complete autonomy for visual servoing across instances and is able to tackle appearance variations, shape variations, and even noisy odometry, which is a challenging task for existing controllers. The complete framework has been deployed and rigorously evaluated on real MAV in the Gazebo simulation environment as well as unstructured outdoor scenarios. In contrast to previous visual servoing approaches, our approach is capable of servoing between large camera transformations, especially between nonoverlapping scenes. Although we have presented the results of our approach only for vehicles, instance invariant visual servoing can be easily extended to other object categories as well. We have further proposed a two-stage cascading-based network architecture, which is able to achieve superior performance for part segmentation over existing approaches. Generalization to novel viewpoints is one of the challenges faced by a majority of deep learning-based frameworks. This becomes even more crucial when the navigation framework relies on deep learning-based perception. We have also proposed refinements hinged upon synthetic data augmentation and the Bayesian fusion to cater for such issues and improve the performance of deep networks from oblique viewpoints.

Finally, we utilize our instance invariant visual servoing framework for the part-based visual inspection of vehicles using a low-cost off-the-shelf MAV. There is a pressing requirement for such application in the automotive industry for vehicle health monitoring, damage assessment, and merchandising. Accomplishing the inspection task autonomously requires understanding the part-based semantics, unlike existing approaches that only provide a 3D reconstruction. The MAV starts with zero prior knowledge about the scene and still it is able to inspect every part of the vehicle using only a monocular camera and noisy odometry. Although, in this paper, we restrict ourselves to the inspection of cars, however, our approach could easily be extended to other objects, such as airplanes and trains and even to humans for search and rescue missions.

ACKNOWLEDGMENTS

We thank TCS research fellowship for financially supporting this study. We also thank Mr. Ravi Shankar Prasad, Mr. Laxit Gavshinde, and Mr. Saket Saurav for allowing us to validate our approach on their vehicles.

ORCID

Harit Pandya http://orcid.org/0000-0002-3842-1949

REFERENCES

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2015). Segnet: A deep convolutional encoder–decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (12), 2481-2495.

Bakthavatchalam, M., Chaumette, F., & Marchand, É. (2014). Photometric moments: New promising candidates for visual servoing. *IEEE International Conference on Robotics and Automation (ICRA), 2014*. IEEE.

Bircher, A., Kamel, M., Alexis, K., Burri, M., Oettershagen, P., Omari, S., & Siegwart, R. (2016). Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots*, *40*(6), 1059–1078.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., & Siegwart, R. (2016). Receding horizon path planning for 3D exploration and surface inspection. *Autonomous Robots*, 1–16.

Blake, A., Rother, C., Brown, M., Perez, P., & Torr, P. (2004). Interactive image segmentation using an adaptive GMMRF model. *Computer Vision-ECCV 2004*, 428–441.

Chaumette, F. (1998). Potential problems of stability and convergence in image-based and position-based visual servoing. In *The Confluence of Vision and Control* (pp. 66–78). Springer.

Chaumette, F. (2004). Image moments: A general and useful set of features for visual servoing. *IEEE Transactions on Robotics*, *20*(4), 713–723.

Chaumette, F., & Hutchinson, S. (2006). Visual servo control. I. Basic approaches. *IEEE Robotics & Automation Magazine*, *13*(4), 82–90.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2016). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(4), 834–848.

Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., & Yuille, A. (2014). Detect what you can: Detecting and representing objects using holistic models and body parts. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1971–1978.

Chin, R. T., & Harlow, C. A. (1982). Automated visual inspection: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6), 557–573.

Crombez, N., Caron, G., & Mouaddib, E. M. (2015). Photometric Gaussian mixtures based visual servoing. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015*, 5486–5491. IEEE.

Dai, J., He, K., & Sun, J. (2016). Instance-aware semantic segmentation via multi-task network cascades. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3150–3158.

Das, J., Cross, G., Qu, C., Makineni, A., Tokekar, P., Mulgaonkar, Y., & Kumar, V. (2015). Devices, systems, and methods for automated monitoring enabling precision agriculture. *IEEE International Conference on Automation Science and Engineering (CASE), 2015*, 462–469. IEEE.

Dryanovski, I., Valenti, R. G., & Xiao, J. (2013). An open-source navigation system for micro aerial vehicles. *Autonomous Robots*, *34*(3), 177–188.

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, *111*(1), 98–136.

Fang, Z., Yang, S., Jain, S., Dubey, G., Roth, S., Maeta, S., & Scherer, S. (2017). Robust autonomous flight in constrained and visually degraded shipboard environments. *Journal of Field Robotics*, *34*(1), 25–52.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(9), 1627–1645.

Fraundorfer, F., Heng, L., Honegger, D., Lee, G. H., Meier, L., Tanskanen, P., & Pollefeys, M. (2012). Vision-based autonomous mapping and

exploration using a quadrotor MAV. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012*, 4557–4564. IEEE.

Furrer, F., Burri, M., Achtelik, M., & Siegwart, R. (2016). Rotors—A modular gazebo mav simulator framework. *In Robot Operating System (ROS)* (pp. 595-625). Cham: Springer.

Gonfaus, J. M., Boix, X., Van de Weijer, J., Bagdanov, A. D., Serrat, J., & Gonzalez, J. (2010). Harmony potentials for joint classification and segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010*, 3280–3287. IEEE.

Hejrati, M., & Ramanan, D. (2012). Analyzing 3D objects in cluttered images. In *Advances in Neural Information Processing Systems* (pp. 593–601).

Hutchinson, S., Hager, G. D., & Corke, P. I. (1996). A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, *12*(5), 651–670.

Kar, A., Tulsiani, S., Carreira, J., & Malik, J. (2015). Category-specific object reconstruction from a single image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1966–1974.

Koenig, N., & Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004 (IROS 2004), 3*, 2149–2154. IEEE.

Kumar, G., Pandya, H., Gaud, A., & Krishna, K. M. (2017). Pose induction for visual servoing to a novel object instance. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017*, 2953–2959.

Li, C., Zia, M. Z., Tran, Q. H., Yu, X., Hager, G. D., & Chandraker, M. (2017, July). Deep Supervision with Shape Concepts for Occlusion-Aware 3D Object Parsing. In CVPR.

Lim, J. J., Pirsiavash, H., & Torralba, A. (2013). Parsing IKEA objects: Fine pose estimation. *Proceedings of the IEEE International Conference on Computer Vision*, 2992–2999.

Lin, Y., Gao, F., Qin, T., Gao, W., Liu, T., Wu, W., & Shen, S. (2018). Autonomous aerial navigation using monocular visual-inertial fusion. *Journal of Field Robotics*, *35*(1), 23–51.

Liu, W., Rabinovich, A., & Berg, A. (2015). ParseNet: Looking wider to see better. arXiv preprint arXiv:1506.04579.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer vision, 1999*, *2*, 1150–1157. IEEE.

Ma, L., Stückler, J., Kerl, C., & Cremers, D. (2017, September). Multi-view deep learning for consistent semantic mapping with rgb-d cameras. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on (pp. 598-605). IEEE*.

Maire, M., Stella, X. Y., & Perona, P. (2011). Object detection and segmentation from joint embedding of parts and pixels. *IEEE International Conference on Computer Vision (ICCV), 2011*, 2142–2149. IEEE.

Maji, S., & Shakhnarovich, G. (2012). Part annotations via pairwise correspondence. *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Morgenthal, G., & Hallermann, N. (2014). Quality assessment of unmanned aerial vehicle (UAV) based visual inspection of structures. *Advances in Structural Engineering*, *17*(3), 289–302.

Murthy, J. K., Sharma, S., & Krishna, K. M. (2017). Shape priors for real-time monocular object localization in dynamic environments. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2017, Vancouver, BC, Canada, September 24–28, 2017*, 1768–1774.

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. *European Conference on Computer Vision*, 483–499. Springer.

Oliveira, G. L., Bollen, C., Burgard, W., & Brox, T. (2018). Efficient and robust deep networks for semantic segmentation. *The International Journal of Robotics Research*, *37*(4-5), 472-491.

Ozaslan, T., Loianno, G., Keller, J., Taylor, C. J., Kumar, V., Wozencraft, J. M., & Hood, T. (2017). Autonomous navigation and mapping for inspection of penstocks and tunnels with MAVs. *IEEE Robotics and Automation Letters*, *2*(3), 1740–1747.

Pandya, H., Krishna, K. M., & Jawahar, C. V. (2015). Servoing across object instances: Visual servoing for object category. *IEEE International Conference on Robotics and Automation (ICRA), 2015*, 6011–6018. IEEE.

Pandya, H., Krishna, K. M., & Jawahar, C. V. (2016). Discriminative learning based visual servoing across object instances. *IEEE International Conference on Robotics and Automation (ICRA), 2016*, 3447–3454. IEEE.

Plath, N., Toussaint, M., & Nakajima, S. (2009). Multi-class image segmentation using conditional random fields and global classification. *Proceedings of the 26th Annual International Conference on Machine Learning*, 817–824. ACM.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems* (pp. 91–99).

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *IEEE International Conference on Computer Vision (ICCV), 2011*, 2564–2571. IEEE.

Schauwecker, K., & Zell, A. (2014). On-board dual-stereo-vision for the navigation of an autonomous MAV. *Journal of Intelligent & Robotic Systems*, *74*(1-2), 1–16.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Shen, S., Michael, N., & Kumar, V. (2011). Autonomous multi-floor indoor navigation with a computationally constrained MAV. *IEEE International Conference on Robotics and Automation (ICRA), 2011*, 20–25. IEEE.

Tang, M., Gorelick, L., Veksler, O., & Boykov, Y. (2013). GrabCut in one cut. *Proceedings of the IEEE International Conference on Computer Vision*, 1769–1776.

Teixeira, L., & Chli, M. (2017). Real-time local 3D reconstruction for aerial inspection using superpixel expansion. *IEEE International Conference on Robotics and Automation (ICRA), 2017*, 4560–4567. IEEE.

Tulsiani, S., & Malik, J. (2015). Viewpoints and keypoints. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1510–1519.

Usenko, V., von Stumberg, L., Pangercic, A., & Cremers, D. (2017, September). Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on (pp. 215-222). IEEE*.

Vicente, S., Carreira, J., Agapito, L., & Batista, J. (2014). Reconstructing PASCAL VOC. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 41–48.

Weiss, S., Scaramuzza, D., & Siegwart, R. (2011). Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments. *Journal of Field Robotics*, *28*(6), 854–874.

Wu, A., Johnson, E. N., Kaess, M., Dellaert, F., & Chowdhary, G. (2013). Autonomous flight in GPS-denied environments using monocular vision and inertial sensors. *Journal of Aerospace Information Systems*, *10*(4), 172–186.

Xiang, Y., Mottaghi, R., & Savarese, S. (2014). Beyond PASCAL: A benchmark for 3D object detection in the wild. *IEEE Winter Conference on Applications of Computer Vision (WACV), 2014*, 75–82. IEEE.

Yu, F., & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122.

Zheng, Y., Sugimoto, S., & Okutomi, M. (2013). ASPnP: An accurate and scalable solution to the perspective-n-point problem. *IEICE Transactions on Information and Systems*, *96*(7), 1525–1535.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., & Torralba, A. (2017). Scene parsing through ADE20K dataset. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Zhou, X., Leonardos, S., Hu, X., & Daniilidis, K. (2015). 3D shape estimation from 2D landmarks: A convex relaxation approach. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4447–4455.

Zia, M. Z., Stark, M., Schiele, B., & Schindler, K. (2013). Detailed 3D representations for object recognition and modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2608–2623.

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of the article.

How to cite this article: Pandya H, Gaud A, Kumar G, Krishna KM. Instance invariant visual servoing framework for part-aware autonomous vehicle inspection using MAVs. *J Field Robotics*. 2019;1–27. https://doi.org/10.1002/rob.21859

APPENDIX

A.1. Index of multimedia extensions

A video presenting further experiments is available as Supporting Information in the online version of this article.

Extension	Media type	Description
1	Video	It presents a visualization of the pipeline and additional experimental trails on MAV

Note. MAV: Micro Aerial Vehicle.

A.2. Additional qualitative results

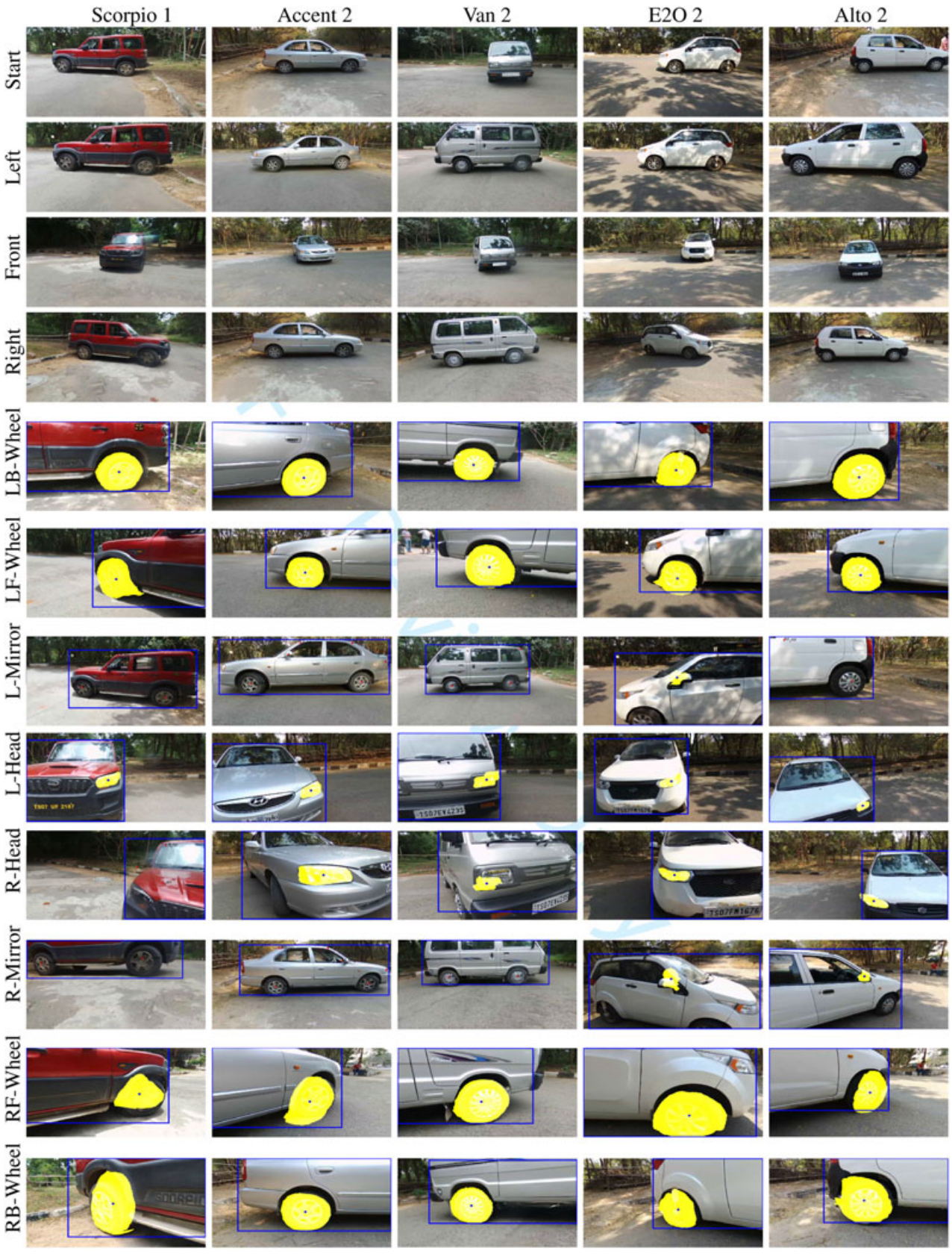

FIGURE A1 More qualitative results for complete autonomous part inspection pipeline in outdoor environments [Color figure can be viewed at wileyonlinelibrary.com]