Pose Induction for Visual Servoing to a Novel Object Instance

Gourav Kumar*[1], Harit Pandya*[1], Ayush Gaud[1] and K. Madhava Krishna[1]

***Abstract*— Present visual servoing approaches are instance specific i.e. they control camera motion between two views of the same object. We address the problem of visual servoing to a novel instance of an object category: given a desired view of any instance (source) of an object category, the robot is required to servo to the corresponding view of another instance (target) from the same object category. We formulate visual servoing across instances as iterative pose induction and pose alignment problem. Here, the desired camera pose of the target (instance on which servoing is done) is induced from a desired view of a source instance (any instance from the same category). Once the desired camera pose is transferred through pose induction, the pose alignment step is solved by estimating the current pose using the semantic reconstruction of target followed by a pose based visual servoing (PBVS) iteration. To tackle large variation in appearance across object instances in a category, we employ visual features that uniquely correspond to locations of object's parts in images. These part-aware keypoints are learned from annotated images using a convolutional neural network (CNN). Advantages of using such part-aware semantics are two-fold. Firstly, it conceals the illumination and textural variations from the visual servoing algorithm. Secondly, semantic keypoints result in more accurate matching compared to local appearance based descriptors like SIFT. We validate the efficacy of our approach through experiments in simulation as well as on a quadcopter. Our approach results in acceptable desired camera pose and smooth velocity profile. We also show results for large camera transformations with no overlap between current and desired pose for 3D objects, which is desirable in servoing context.**

I. INTRODUCTION

Visual servoing utilizes image sensory information to move a robotic system towards a goal position with respect to the given object. 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 [1]. Traditionally visual servoing approaches are classified into two high-level streams based on how the control objective is defined [2]. Position based visual servoing (PBVS) utilizes visual features to estimate the object's pose in robot's Cartesian space. Estimating robot's pose requires additional information regarding the geometry of the object which is obtained by either explicit knowledge of object's 3D model or by reconstructing the object while servoing. Whereas, image based visual servoing (IBVS) directly controls the robot in image space. However, IBVS camera trajectory is not explicitly controlled in the Cartesian frame as a result controller could lead to a local minima [3].

[1] International Institute of Information Technology, Hyderabad, India {harit.pandya,gourav.kumar}@research.iiit.ac.in, ayush.gaud@gmail.com, mkrishna@iiit.ac.in

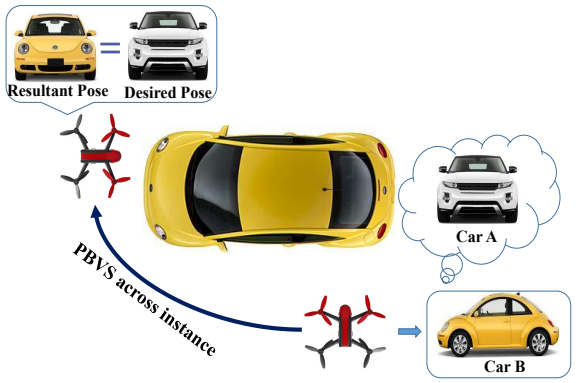
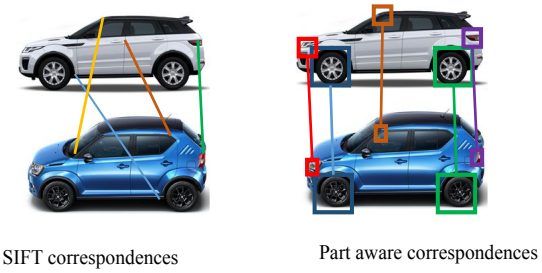


Fig. 1. **Aim:** Given a desired view of an object instance (source) and its 3D model, we consider the problem of servoing to the corresponding desired view of a different instance (target) using a quadcopter. In this figure, the quadcopter is initially observing the target from side view. The servoing task requires the quadcopter to move to the frontal view of the target so that observed view matches the given desired view.

For visual servoing quality of features have a strong correlation with the performance of a visual servoing approach. Previously, low-level geometric primitives (for examples corners, lines, contour) were proposed as visual features [1]. However, extracting these features reliably is non-trivial. Thus, a few approaches used local appearance based features like SIFT [4]. The issue with such features is that they only account for local variations, therefore the matching accuracy is compromised. In this paper, we propose part-aware keypoints as visual features that encode the global perspective of the object and retain only the meaningful precise keypoints as shown in figure 2. Furthermore, these keypoints describe the locations of object's parts that could be related semantically to object, thus they provide a unique and accurate matching across instances in contrast to local descriptors such as SIFT, ORB etc. We trained a convolutional neural network (CNN) based on stacked hour-glass architecture [5] to learn these part-aware keypoints on Pascal 3D dataset [6].

Visual servoing controller defines an error function between current and desired configuration of visual features. Since these visual features are geometrically related, a possible solution exists in $SE(3)$ space such that the error is regulated to zero at the desired pose. However, while servoing across instances in the same object category, the geometry of the objects does not permit the error to diminish at the desired pose. This limits the scope of traditional visual servoing to views to same object. These problems arise frequently in practical scenarios where servoing to a new instance is required, especially in the case of manipulation

Preprint submitted to 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received March 1, 2017.



SIFT correspondences Part aware correspondences

Fig. 2. SIFT vs Part-aware keypoints: SIFT based keypoints might result in an incorrect matching when the instances are different, whereas our part-aware semanticsare more suitable for computing correspondences across object instances.

or autonomous navigation in unknown environments. In this work, we leverage visual servoing to a novel previously unseen object instance (target) from a known object category. Figure 1 describes a use-case scenario in which license plate inspection is being done using a drone (quadcopter). Given the desired pose of the front of car A (white car: source instance) from which number plate is visible, the drone is required to servo to the front of car B (yellow car: target instance), which is a novel instance for the drone. We formulate servoing to a novel instance as iterative pose induction and alignment problem. The pose induction step requires inferring the desired pose with respect to target from the given desired pose with respect to the source. This is achieved through actively moving the drone and simultaneously reconstructing the target instance (car B) using our part-aware semantic features. The target instance is reconstructed in camera frame while the source is given in the world frame. Therefore, a transformation is required between camera and world frames, which unfortunately is not always available. Hence, we define a canonical frame where axis and origin are defined in a semantic sense. Following the alignment rules of the canonical frame, both source and target are transformed to this canonical frame. Now, the desired camera pose could be directly transferred from source to the target since they both are in the same frame. Once the desired pose and current pose refer to the target instance, the pose alignment step can be solved by a PBVS iteration. Both pose induction and alignment steps are solved in part-aware semantic space so that reconstruction is accurate and requires fewer computations.

Contributions: Our primary contribution is a pose induction framework for visual servoing to a novel object (target) instance in a given desired pose using just one standard model (or source instance) of that category as the basis. Our framework is able to tackle large variations in appearance and illumination through part-aware keypoint prediction learned using a CNN. Secondly, our semantic reconstruction of the object is more precise and fast, thanks to accurate and sparse part-aware keypoints. Thirdly, we propose a context based alignment scheme that could cater for large variations in shape by using a canonical frame and an associated alignment protocol. Finally, we showcase results of our framework for large camera transformations in simulation as well as on quadcopter.

A. Related work

Classical visual servoing approaches used geometrical primitives like keypoints, lines, contours etc. as visual features [1]. However, reliable extraction of such visual features from real images in itself a challenging task. Using local keypoint descriptors like SIFT, SURF and ORB as visual features [4], results in erroneous matching which degrades the performance of visual servoing approaches especially when the instances are different. In recent years, information theoretic visual features like pixel intensities [7], image gradient [8], histogram of intensities [9], image moments [10] etc. have been rigorously explored for visual servoing tasks. The advantage of using these data driven features is that tracking individual keypoints is no longer required as statistics of the current view is matched to that of the desired view which results in robust and precise alignment. However, the convergence domain is compromised, since current and desired views are no longer geometrically related. Hence, for large camera transformations the pixel intensity values might not correspond to desired view pixel intensities. Also, photometric and its derivative approaches are sensitive to illumination variations. 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 computer vision. Previous approaches from computer vision literature report superior performance in keypoints matching when the keypoints were conditioned on object category especially when the keypoints were semantically related to object's parts [11],[12]. Motivated from recent breakthroughs in CNNs, Tulsiani et al. [13] presented a CNN that was able to learn part-aware keypoints through supervision. Recently [5] proposed a stacked hour-glass architecture for CNN showcased state-of-art results for keypoint prediction on humans. In this paper, we trained a stacked hour-glass CNN on a combined dataset composed of Pascal 3D [6] dataset for cars. We obtained superior results for keypoints detection over [13] for 'car' object category.

The problem of visual servoing across object instances was first introduced by Pandya et al. [14]. They used part-aware keypoints for making the approach robust to textural variation across instances in an object category. They 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 pre-rendered poses for every visual servoing iteration, which makes the approach computationally expensive. A discriminative learning based framework was also proposed for visual servoing across instances [15]. Where, authors proposed principal orientation glyph (POG) as visual features and a classification error based controller was used for achieving geometry invariance. However, their interaction matrix was numerically computed,

which resulted in a relatively smaller convergence domain. In this paper, we employ a novel pose induction framework that is able to borrow pose of a source object instance and estimate how another instance (target) will look in this view.

This pose induction provides geometric invariance to our approach and is used to transfer the desired pose to the target instance. We further employ PBVS controller to attain the estimated desired pose with respect to the target instance.

II. Problem formulation

We consider a global world frame $\mathcal{F}_o$, the current camera frame $\mathcal{F}_c$, the desired camera frame $\mathcal{F}_{c^*}$ and a reference frame attached to the target object $\mathcal{F}_X$. We further assume that the object category has been previously seen by the robot and a 3D model of same object category with annotated 3D semantic keypoints Y is available. The problem of servoing to a novel target instance (X in this case) requires moving the camera from current pose $\mathcal{F}_c$ to the desired pose $\mathcal{F}_{c^*}$, where $\mathcal{F}_{c^*}$ needs to be estimated from the desired image I_Y^* and its corresponding model Y.

III. Overall pipeline

We have divided the task of across instance visual servoing into two sub-modules namely, pose induction and pose alignment. The task of pose induction sub-module is to obtain the desired pose $\mathcal{F}_{c^*}$ from the given image I_{c^*} and its corresponding annotated 3D model Y. We employ a CNN to compute the part-correspondences x and y from current and desired images I_x and I_y^* respectively. Once we obtain the projections y of source instance we use its 3D model to compute the desired camera pose $\mathcal{F}_{c^*}$ by solving the perspective-n-point (PnP) problem. The $\mathcal{F}_{c^*}$ computation is required only once and hence could be performed offline.

Our framework performs real-time simultaneous reconstruction and servoing of the target instance in a closed loop fashion. The pipeline starts with obtaining current image I_x from the robot which is passed on to our keypoint prediction network. Note that the reconstruction is in a camera frame and $\mathcal{F}_{c^*}$ is in a global frame. Hence, we align both $\widehat{X}$ and Y in a single virtual canonical frame $\mathcal{F}_v$, so that the desired pose of the camera with respect to the target instance could be estimated from $\mathcal{F}_{c^*}$ through pose induction. The induced pose is then passed on to pose alignment sub-module, which relies on PBVS controller for generating the velocity commands for the quadcopter. These velocity commands are then tracked using the local controller of the robot. The entire pipeline is repeated iteratively till PBVS control error is within acceptable limits. This pipeline is summarized in figure 3.

A. Keypoint Prediction

In visual servoing classical approaches used keypoints for visual servoing, however computing keypoints which provide accurate correspondences is still an open problem for researchers. Previous approaches considered local descriptors like SIFT, ORB etc. for computing keypoints [4], however, these keyponts do not consider a global view of the image as result the matches are not reliable. In this work, we aim to compute keypoints that are not only robust to illumination and pose variations but also consistent across different instances. Motivated by how humans register any object as an ensemble of parts with some semantics (for example we see car as a composition of parts like headlight, wheels, mirrors etc.), computer vision approaches try to learn the location of object's parts in a given image through supervised machine learning approaches such as support vector machines [16]. Motivated from breakthroughs in deep learning, Tulsiani et al. [13] presented a CNN based architecture for learning part-aware keypoints from images. They reported a significant improvement in keypoint prediction accuracy by conditioning keypoints inference using viewpoint estimations. They proposed two different networks for keypoint predictions both at coarser and finer scales. However, their approach requires running three neural networks for a single image, which makes their implementation slower and computationally expensive.

In this work, we leveraged the more recent hourglass architecture deep convolutional neural network for our task of keypoint prediction. This network architecture was initially proposed by Newell et al. [5] for human pose estimation. The design of hourglass network captures information at multiple scales similar to [13] in a single network, as a result, it is faster and more accurate compared to [13]. Use of stacked hourglass in a way provides an end-to-end solution for estimating part-aware keypoints. We trained a deep convolutional network composed of eight hourglass module figure 4 stacked one after the other. The highest resolution of the hourglass is 64x64. The full network starts with a 7x7 convolutional layer with a stride 2 followed by residual module and a max pooling which brings down the resolution from 256x256 to 64x64. The stacking of hourglass modules assures both repeated bottom-up and top-down re-evaluation of initial feature estimates. This network was trained on annotated car models from pascal 3D dataset [6] using Stochastic Gradient Descent (SGD) with a Euclidean loss. The prediction accuracy of our network was approximately 93% with a tolerance of two pixels, which is better compared to 81.3% on cars claimed by Tulsiani et al.[13]. The dataset was annotated for fourteen different keypoints and it gives the confidence scores along with the image coordinate corresponding to each keypoints for the given image. The scores corresponding to occluded parts or less guessable parts show a clear diminish which helped us to filter out the less confident predictions by the network. These predictions were further used in the pipeline as features for reconstruction and error calculation for visual servoing.

B. Desired Pose Estimation

The problem of determining pose of a calibrated camera from *n* correspondences between 3D reference points and their 2D projections are known as "*Perspective-n-Point problem*"(*PnP*). The solution to this PnP problem is used for estimating the camera extrinsics (*R,t*). Given 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

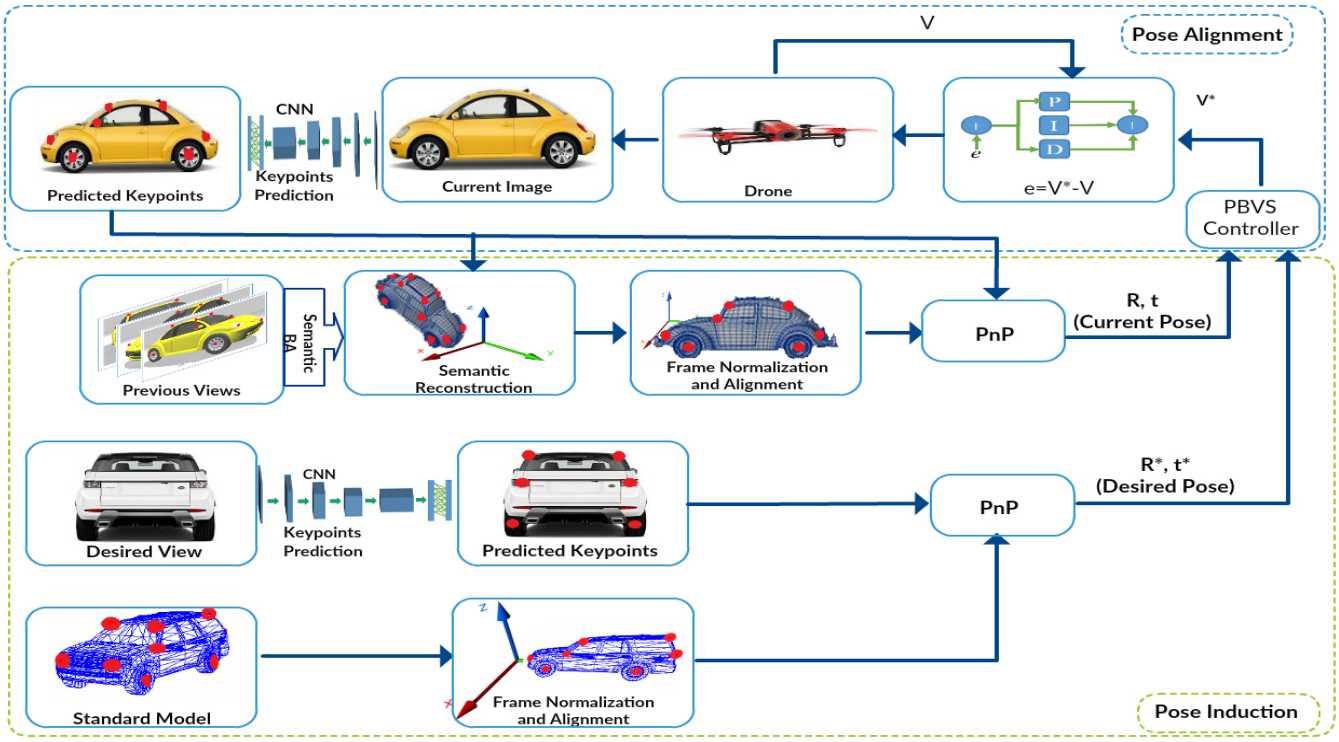


Fig. 3. **Overall pipeline of the proposed approach.** Pose Induction: given the desired view of a source instance of a car in form of an image, our deep network predicts the keypoints which, along with source instance model information, is used to predict the desired transformations ($\mathcal{F}_c^* = (R^*, t^*)$) in a global frame. The keypoints predicted on the current image along with the previous views is used for semantic reconstruction using Bundle Adjustment (BA). This reconstruction is then normalized and aligned with respect to the global frame which is then used along with desired transformations to obtain target instance specific desired pose using perspective-n-point (PnP) solver. This pose is fed to the pose alignment sub-module that employs pose based visual servoing (PBVS) controller for generating control commands for the quadcopter.

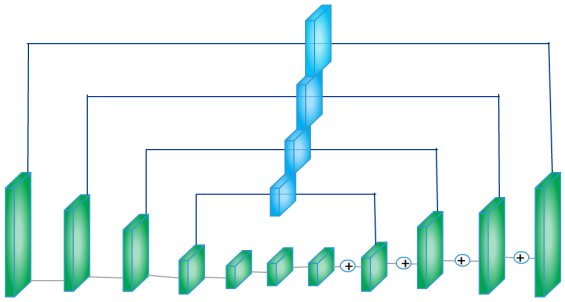


Fig. 4. **An hourglass module:** Each box in the figure corresponds to a convolutional layer. The skips at different scale are used for preserving the spatial information at that resolution.

re-projection error as:

$$\min_{R,t} \sum_{i=1}^{n} ||K(RX_i + t) - x_i||^2$$

subject to

$$R^T R = I$$

PnP is a well established problem in 3D geometry and there are multiple solutions to the problem. We are particularly interested in using ASPnP since it solves the 3D-to-2D correspondences using a Gröebner basis solver, which guarantees a globally optimal camera pose. In this work, we use ASPnP [17], to determine the desired pose of the camera $\mathcal{F}_{c^*}$ from the keypoints of the desired view x using its corresponding 3D annotated model X. We also use ASPnP for estimating the current camera pose with respect to the reconstructed model.

C. Semantic Reconstruction

This approach has been referred to as semantic reconstruction because here features used for reconstruction are predicted keypoints which uniquely corresponds to parts of a vehicle. This step starts with a stereo initialization i.e. giving a translation orthogonal to the optical axis of the camera to form a stereo image pair. Assuming the knowledge of camera parameters, the two frames obtained are used in triangulation for estimating 3D coordinates of the keypoints. The triangulated points are obtained in a frame fixed to the initial camera pose. From the third frame onwards newly obtained keypoint is concatenated to the reconstructed target 3D model after triangulation using current and previous frames. The predicted keypoints along with the partially reconstructed model are also used for getting the position and orientation of the camera using ASPnP [17]. As the method of incremental tracking and mapping of keypoints are prone to drift, hence we use bundle adjustment on the camera transformations available till that instant. This helps us to reduce the error of both reconstructed keypoints structure as well as position and orientation information of the camera frames.

D. Normalization and alignment to canonical frame

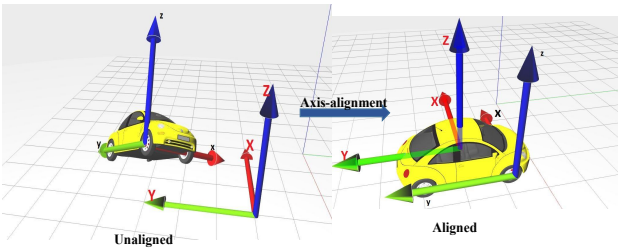


Fig. 5. **Axis-alignment:** The axis of the car frame is aligned parallel to the canonical frame while the car model is centered on the canonical frame origin.

The reconstruction of target instance $\widehat{X}$ by structure from motion (SFM) pipeline is in the initial camera frame $\mathcal{F}_{\widehat{X}}$. While the desired camera pose given by PnP is in the frame of source instance $\mathcal{F}_Y$. Therefore, to compute the desired camera pose with respect to $\widehat{X}$, we need a transformation between $\mathcal{F}_{\widehat{X}}$ and $\mathcal{F}_Y$, which unfortunately is not always available. This is a common problem encountered by almost all SFM based approaches since the transformation between camera frame and world frame is generally unavailable.

We tackle this issue by defining a canonical frame $\mathcal{F}_v$ and transforming (aligning) both $\mathcal{F}_{\widehat{X}}$ and $\mathcal{F}_Y$ to $\mathcal{F}_v$ using an alignment protocol. A valid alignment protocol requires exactly four rules, one rule to define an origin and two rules to define the alignment of any two coordinate axes. Our transformation protocol complies with and feasible because the keypoints have semantic meaning associated with them. For example, consider the transformation of a target instance from $\mathcal{F}_Y$ to $\mathcal{F}_v$. Further, consider a set of four keypoints that represent following part-aware semantics "left front wheel", "right front wheel", "left rear wheel" and "right front wheel". A valid alignment protocol could then be defined as: (i) define origin of $\mathcal{F}_v$ as centroid of the four wheels, (ii) x-axis of $\mathcal{F}_v$ should parallel to a ray from "left rear wheel" and (iii) y-axis of $\mathcal{F}_v$ should be parallel to a ray from "left front wheel" to "right front wheel". To maintain homogeneity in the scale the models are further normalized after alignment. Even though exact transformation between $\mathcal{F}_Y$ and $\mathcal{F}_v$ is not known, it is still possible to align them based on a given transformation protocol, this is only possible because our keypoints have a semantic meaning associated with them. The procedure of aligning a 3D model to our canonical frame is shown in figure 5.

E. Pose alignment using PBVS

After projecting the keypoints in desired view and predicting the coordinates in the current view, we employ PBVS controller to servo between two views of the target instance. Similar to classical PBVS, we consider $\mathcal{F}_c$ as current camera frame, $\mathcal{F}_{c^*}$ as desired camera frame and frame $\mathcal{F}_o$ attached to the object. We further define translation vector ${}^{c^*}\mathbf{t}_c$ and rotation matrix ${}^{c^*}\mathbf{R}_c$ as translation and rotation of $\mathcal{F}_{c^*}$ with respect to $\mathcal{F}_c$. The PBVS controller is then designed using current feature vector $\mathbf{s} = ({}^{c^*}\mathbf{t}_c), \theta\mathbf{u})$. In that case, we have $\mathbf{s}^* = 0$, $\mathbf{e} = \mathbf{s}$ that results in following control law:

$$\mathbf{v}_c = -\lambda {}^{c^*}\mathbf{R}_c^T {}^{c^*}\mathbf{t}_c \quad (1)$$

$$\omega_c = -\lambda\theta\mathbf{u}. \quad (2)$$

IV. Experiments and Results

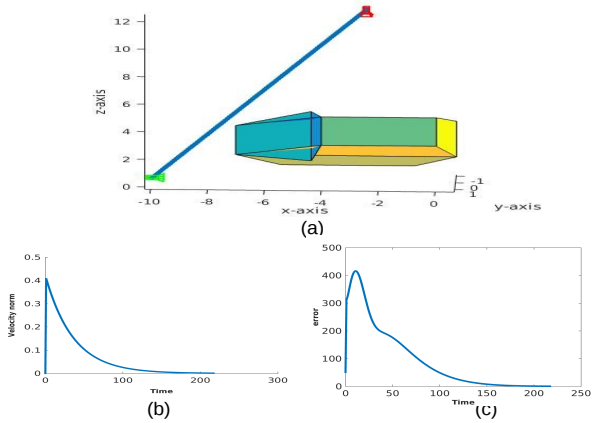


Fig. 6. **Servoing to front:** *(a)* Camera trajectory- Red:initial pose; Green:Final/Desired pose; *(b)* Camera Velocity; *(c)* Error between current and desired pose

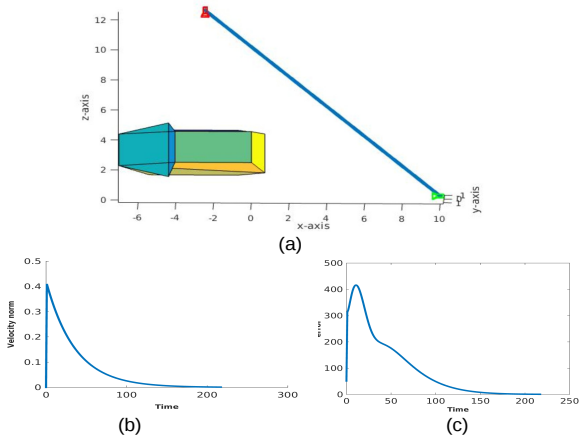


Fig. 7. **Servoing to back:** *(a)* Camera trajectory- Red:initial pose; Green:Final/Desired pose; *(b)* Camera Velocity; *(c)* Error between current and desired pose. Note the straight line trajectory beacuse of PBVS controller with zero noise. Also notice how the velocity and error exponentially decrease to zero.

We evaluated the proposed approach with a wide range of camera transformations between initial and desired poses in three stages namely, synthetic data, simulation in gazebo environment and with a quadcopter. Experiments on synthetic data helps us to experimentally verify the entire pipeline in ideal conditions i.e. when the keypoints are manually annotated. We then elevate the experiments on synthetic data

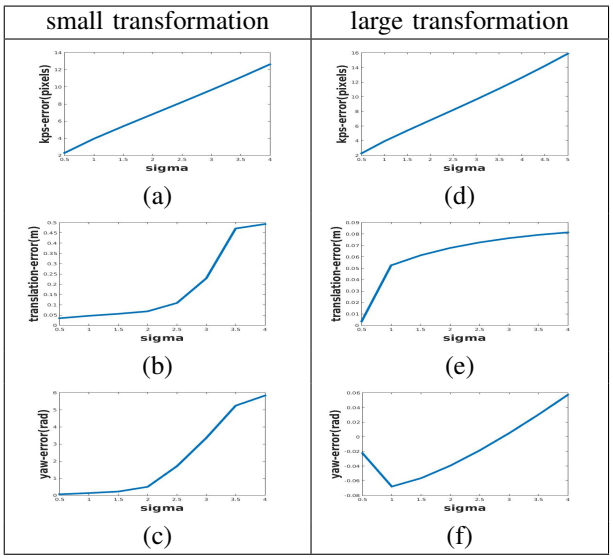


Fig. 8. **Synthetic results with additive Gaussian noise.** Experiment considering additive Gaussian noise with increasing covariance in the keypoint prediction is presented. *(a),(d):* Average of keypoints error between final and desired pose; *(b),(e):* Error between desired and final camera position;*(c),(f):* Error between desired and achieved yaw angles. The error in rotation and translation increases as the variance in gaussian increases. Note that the x-axis starts from standard deviation=0.5.

by using additive Gaussian noise in keypoints measurements and analyzing the convergence properties. For the next set of experiments, we aim to qualitatively analyze the efficacy of our approach using Gazebo framework [18]. Gazebo helps us to render a 3D model from a given viewpoint similar to a real environment. It also simulates real world physics and also enables variations in lighting conditions. In this experiment keypoints were predicted on rendered images using our keypoint network. We finally showcase servoing results for a real car using a quadcopter. Our code for visual servoing on synthetic data and the weights of our trained keypoint network is available at the project webpage [1].

A. *Experiment on synthetic data*

For this experiment, ten 3D models of cars were annotated with 14 keypoints. As the models were of different scales, as an initial step, keypoints were centered and normalized. The evaluation was performed using two different set of 3D points to simulate car keypoints. A random view projection from the first set was used as the desired view while the second set was used for projection from target instance. (figures 6 & 7) shows the efficacy of the proposed algorithm by servoing successfully despite the large transformation between the initial pose and the desired pose. For better visualization, a 3D model made of polygons with keypoints as its vertices are added in the simulation. Results show error and velocity decreasing exponentially with time, which is desirable for visual servoing task.

To test the robustness of our approach, Gaussian noise generated with a range of covariance was added to the projected keypoints. Quantitative results of this experiment is reported in Figure 8. As can be seen in the graphs, the convergence error in terms of yaw error, translation error and feature (keypoints) error, all increases with increasing value covariance of the added Gaussian noise. The experiment show similar trend for both small and large desired camera transformations.

Previously harit et al.[14] used a linear combination of 3D models to estimate target. This results in reconstruction errors even without considering noise. Authors report an average 10 % noise in translation and rotation. However, since we are explicitly reconstructing the target and transferring the desired pose from source after axis alignment, hence in the absence of noise we achieve no translational and rotational error.

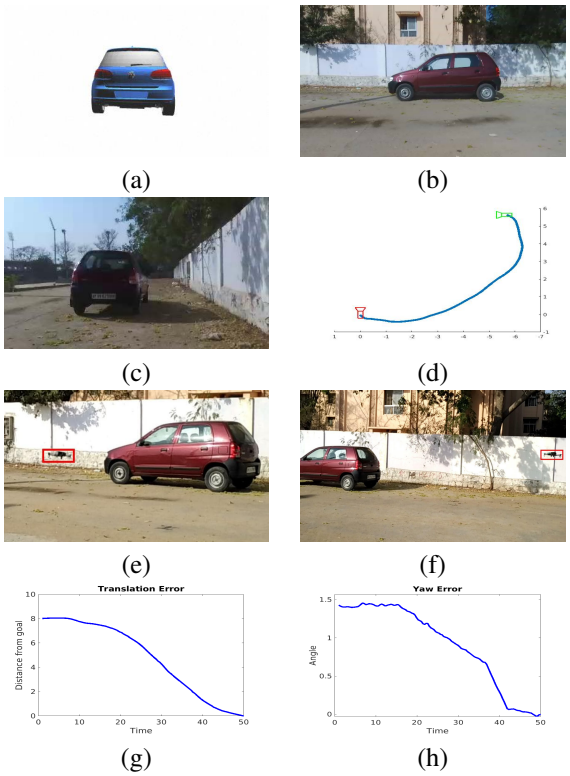


Fig. 9. **Quadcopter experiment** Experiment with a quadcopter on a real car in an outdoor environment. *(a):* Desired view of a standard car(source instance); *(b):* Current view of the car(target instance); *(c):* Final view of the car (target instance); *(d):* 3D trajectory taken by the quadcopter;*(e):* Starting pose of the quadcopter (shown in red box); *(f):* Final pose of the quadcopter (shown in red box); *(g):* Translational error with time; *(h):* Yaw error with time. Note that although we are using PBVS controller the trajectory is curved due to noisy keypoints and reconstruction error. Also, notice the gradual decrease in translation and rotation error as the approach proceeds.

B. *Qualitative Results on* Gazebo *simulation:*

Second we performed the simulation in using a quadcopter model in an environment with car a model in Gazebo [18] with Robot Operating System (ROS). We tested the simulation with varying car models, desired transformations and lighting conditions. The performance of the simulation is shown in figure 10. Second column of the figures shows the desired view in the source model. The last column represents the error between the desired view and the final

[1] http://robotics.iiit.ac.in/urls/vs_induction

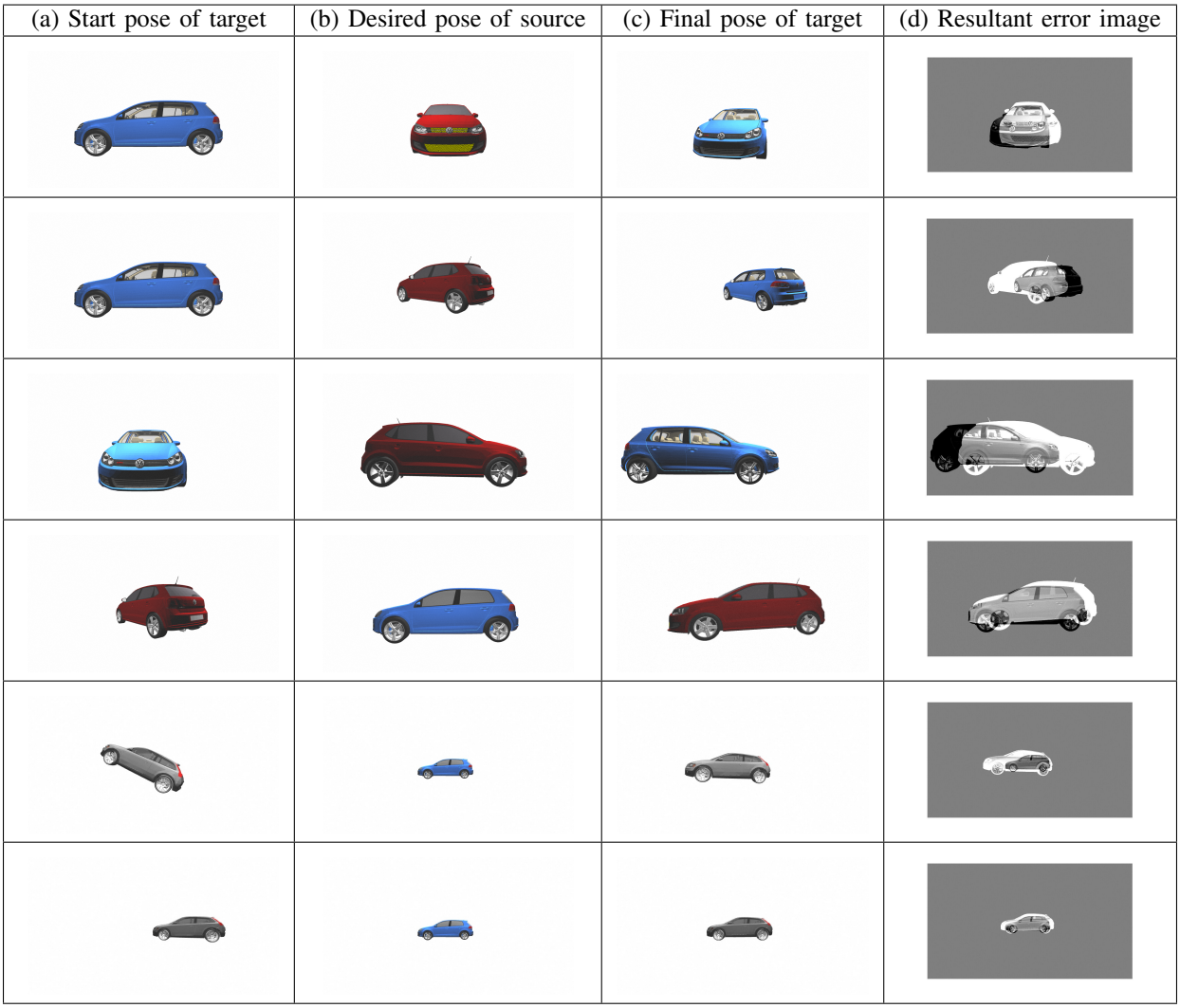


Fig. 10. **Qualitative results.** (a) Initial pose captured by the robot. (b) Desired camera pose of a different instance. (c) Resultant positioning of camera achieved by our approach. (d) Resultant error image. Note the similarity in the resultant pose achieved by the proposed approach compared to the desired pose provided, even for large camera transformations.

view captured by the quadcopter.

C. Real world experiment

Lastly, as a final proof of concept and viability of our approach, we tested over real quadcopter with a monocular camera and local velocity controller with a car in an outdoor environment figure 9. In this experiment, we evaluate our approach on real world scenarios using a Parrot Bebop drone. Since, quadcopters are under-actuated, only 4 DOF tasks were selected for visual servoing. In real world, it is difficult to accurately predict the position of a drone. Hence, we report the qualitative results and an approximate trajectory generated and reported by the drone by fusion of inertial measurement unit (IMU), sonar sensor and optical flow sensor facing downward. Again, the transformation between the initial and the desired pose is large. The quadcopter local controller tracks the velocity commands generated by our PBVS controller. The keypoints are predicted by our CNN network. As our CNN takes the bounding box of the car along with the image, we run a separate object detection network YOLO (you only look once) by Redmon et al. [19] in parallel. The CNN forward pass for keypoint prediction as well as object detecton was performed using a laptop computer with Core i7 CPU, Nvidia Quadro M2000M GPU and 16 GB RAM. It took 140 ms for one forward pass to complete on the machine. The image captured by the drone and corresponding control commands generated by the PBVS controller were exchanged between the system and drone over wifi channel. The robustness of this system made it possible to conduct successful outdoors experiment.

V. CONCLUSION

In This work, we have introduced a novel pose induction and alignment pipeline for across instance visual servoing for an object category. We also have trained a CNN that is able to achieve state-of-art results for semantic keypoint predictions on vehicles. We evaluated our approach though various experiments on synthetic data, in Gazebo simulation

environment as well as on actual quadcopter. Our approach is able to achieve acceptable camera pose even for large initial transformations in camera pose and high intra-category variation in appearance and shape among object instances. Although we have tested our approach only for vehicle class objects, this approach can be easily extended to other object categories as well. The motivation behind selecting car as object category was to automate vehicular inspection. We have made our code publicly available for further contribution.

REFERENCES

[1] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," in *MRA*, 2006.

[2] P. Rives, "Visual servoing based on epipolar geometry," in *IROS*, 2000.

[3] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The confluence of vision and control*, 1998.

[4] F. Hoffmann, T. Nierobisch, T. Seyffarth, and G. Rudolph, "Visual servoing with moments of sift features," in *SMC*, 2006.

[5] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*. Springer, 2016.

[6] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond pascal: A benchmark for 3d object detection in the wild," in *(WACV)*, 2014.

[7] M. Bakthavatchalam, F. Chaumette, and É. Marchand, "Photometric moments: New promising candidates for visual servoing," in *ICRA*.

[8] E. Marchand and C. Collewet, "Using image gradient as a visual feature for visual servoing," in *IROS*, 2007.

[9] A. Dame and E. Marchand, "Mutual information-based visual servoing," in *TRO*, 2011.

[10] F. Chaumette, "Image moments: a general and useful set of features for visual servoing," in *TRO*, 2004.

[11] S. Maji and G. Shakhnarovich, "Part annotations via pairwise correspondence," in *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[12] F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," in *PAMI*, 2010.

[13] S. Tulsiani and J. Malik, "Viewpoints and keypoints," in *CVPR*, 2015.

[14] H. Pandya, K. M. Krishna, and C. V. Jawahar, "Servoing across object instances: Visual servoing for object category," in *ICRA,2015*.

[15] H. Pandya, K. M. Krishna, and C. Jawahar, "Discriminative learning based visual servoing across object instances," in *(ICRA)*. IEEE, 2016.

[16] L. Bourdev and J. Malik, "Poselets: Body part detectors trained using 3d human pose annotations," in *ICCV*, 2009.

[17] Y. Zheng, S. Sugimoto, and M. Okutomi, "Aspnp: An accurate and scalable solution to the perspective-n-point problem," *IEICE TRANSACTIONS on Information and Systems*, 2013.

[18] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *(IROS)*. IEEE, 2004.

[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

Preprint submitted to 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received March 1, 2017.