2018 15th Conference on Computer and Robot Vision

Gaze Selection for Enhanced Visual Odometry During Navigation

Travis Manderson[1] and Andrew Holliday and Gregory Dudek

***Abstract*— We present an approach to enhancing visual odometry and Simultaneous Localization and Mapping (SLAM) in the context of robot navigation by actively modulating the gaze direction to enhance the quality of the odometric estimates that are returned. We focus on two quality factors: i) stability of the visual features, and ii) consistency of the visual features with respect to robot motion and the associated correspondence between frames.**

We assume that local texture measures are associated with underlying scene content and thus with the quality of the visual features for the associated region of the scene. Based on this assumption, we train a machine-learning system to score different regions of an image based on their texture and then guide the robot's gaze toward high scoring image regions.

Our work is targeted towards motion estimation and SLAM for small, lightweight, and autonomous air vehicles where computational resources are constrained in weight, size, and power. However, we believe that our work is also applicable to other types of robotic systems. Our experimental validation consists of simulations, constrained tests, and outdoor flight experiments on an unmanned aerial vehicle. We find that modulating gaze direction can improve localization accuracy by up to 62 percent.

I. INTRODUCTION

This paper describes an approach for enhancing the accuracy and reliability of visual odometry for robot vehicles - especially Unmanned Aerial Vehicles (UAVs) - by actively controlling the vehicle's gaze independently of its position control. Robust and accurate visual odometry is a critical requirement in many real-world robotic applications. The demands placed on such systems are especially stringent because UAVs are often small, lightweight, fragile, and fast-moving. While optical flow computation, ego-motion estimation, and visual odometry (which are highly inter-related ideas) have all been actively investigated for over 40 years with numerous impressive successes, motion estimation that is truly accurate, reliable, and real-time-capable has remained an elusive and challenging goal [1], [2]. In this paper, we provide evidence that the performance of existing real-time-capable Simultaneous Localization And Mapping (SLAM) systems can be improved by allowing the robot to choose certain parts of the scene to estimate odometry without sacrificing the system's ability to run in real time on a UAV.

Several authors have described approaches to ego-motion estimation, but almost all of these approaches are subject to occasional (or more than occasional) failures as a function of the environment's appearance. For example, stereo depth estimation fails when visible structures are very far away compared to the stereo baseline. Using a combination of estimates can allow them to compensate for each other's deficiencies. However, even with such fused estimators, unfavorable viewpoints such as those containing little or no visual texture can cause odometry to fail.

In this work, we attempt to address the shortcomings of existing odometry approaches by shifting the gaze of the UAV-mounted camera during a navigation task, directing the camera to regions that will yield the most promising visual feedback for odometry. To select the best gaze direction, we must consider not only the quantity of visual features that can be observed in each direction, but also their quality: whether they are likely to be stable, geometrically consistent, and observable in subsequent frames. A relevant scenario is illustrated in Fig. 1 where looking at open snow leads to much more unreliable features than looking at the trees. We wish to look in the direction that offers the best *expectation* of high-quality features.

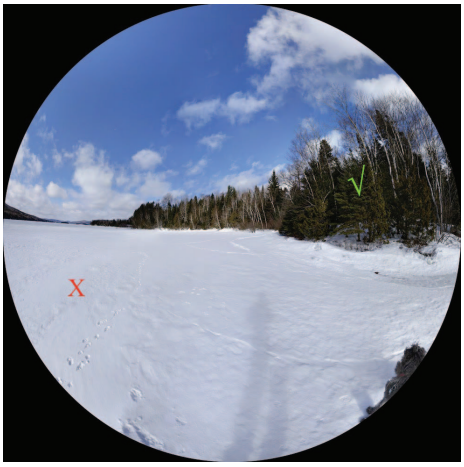

Fig. 1. A typical winter scene where looking at the trees to the right leads to much better motion estimates (on a windless day) than looking at the featureless snow to the left.

Our system uses a learned mapping from scene appearance to robustness for motion estimation and exploits the ability of our UAV to decouple the gaze direction of the camera (and thus the body attitude) from the direction of flight. We build the link between feature quality and visual content by characterizing regions of the scene in terms of their textures and then learning the conditional dependency between texture and the past stability of the associated scene features in an environment-dependent fashion. This approach allows us to learn, for example, that textures associated with leafy trees provide unreliable features on windy days but reliable features on days when the wind is calm.

[1]The authors are part of the Center for Intelligent Machines and the School of Computer Science, McGill University, Canada. This work was supported by NSERC through funding for the NSERC Canadian Field Robotics Network. travism@cim.mcgill.ca

978-1-5386-6481-0/18/$31.00 ©2018 IEEE
DOI 10.1109/CRV.2018.00025

For our texture descriptors, we develop a technique that we previously exploited in preliminary studies [3] with a stereo camera using Local Binary Patterns (LBPs) [4]. LBPs are a well-established basis set for texture classification [5] and their performance is comparable to more classical descriptors such as Gabor functions. They also have the advantage of being very computationally efficient since they can be computed using a small number of arithmetic operations and look-up tables, thus making them appropriate for our real-time, flight-capable implementation.

Our previous work [3] showed that such an approach can be used to predict useful gaze directions and subsequently control the gaze of a pan/tilt-unit-mounted camera on a ground robot following a fixed path indoors under human control. This approach led to an increase in robustness of visual odometry, preventing tracking failure (getting "lost"). The current paper presents an evaluation that is more rigorous and realistic in the following ways: 1) we dispense with the fixed-path constraint in favour of performing real-time trajectory estimation and closed-loop control, 2) we perform our experiments in challenging real-world outdoor environments and in a high-fidelity simulated outdoor environment, and 3) we deploy the technique alongside visual SLAM in real-time on a working UAV.

The rest of the paper is organized as follows: Section II discusses work upon which we build or that is similar to ours. Section III describes the visual SLAM, texture-based patch classification, and gaze control components of our system. Section IV quantitatively validates our system in extensive free-flight simulations and on outdoor fixed-path traversals, followed by a qualitative validation of its effectiveness in free flights outdoors. Finally, in Section V, we summarize our results and consider directions for future research.

II. Related Work

Our work is related to the enhancement of localization and mapping based on data from a vision system. As such, it depends on both estimates of the robot's motion as well as the arrangements of features in the environment. Ego-motion estimation goes back to the seminal work of Horn and Schunck [1] that was later refined over decades. The notion of optical flow as a cue to motion estimation has even deeper roots [6]. Although various authors have considered combinations on ego-motion and scene reconstruction, a modern landmark in using motion for SLAM was the work of Davison [7]. The visual odometry component of our work is related to other modern motion estimation methods such as probabilistic tracking and mapping (PTAM) [8], multi-camera PTAM (MCPTAM) [9], Semi-direct Visual Odometry [10], ORBSLAM [11], recent work by Shen [12], and other similar methods [13], [14].

All these methods perform some sort of feature-based tracking to estimate a system's pose and orientation with respect to a world-fixed frame of reference. In terms of combining visual cues with inertial data (as we do in our SLAM system), several authors have considered how to do this task efficiently and robustly while minimizing the reconstruction error [15], [16], [17], [18]. Despite this corpus of impressive algorithms, the fundamental fact is that the appearance of the scene in the gaze direction has a dominant effect on the ability of a visual odometry system to operate effectively. Vision-based SLAM is inevitably subject to issues of robustness and stability when the visual environment is insufficiently rich.

Work by Peretroukhin [19] showed that the angle between a vehicle's gaze and motion directions has a significant effect on the accuracy of visual odometry pose estimation but did not consider the effects of scene content on different gaze directions. MCPTAM [9] tried to overcome some limitations of PTAM [8], which relied on a fixed-forward camera gaze, by using multiple fixed cameras to track features in multiple non-overlapping views. While effective, processing images from multiple cameras significantly increases the algorithm's computational cost unless image resolution is reduced, which tends to reduce odometric accuracy and robustness.

In recent work by Wu et al. [20], selecting between a pair of fixed alternative cameras based on apparent scene depth has been shown to substantially improve performance, thus demonstrating the value of actively modulating gaze direction even in a binary manner. Our system also exploits gaze shifting, but allows for arbitrary gaze shifts based on appearance content rather than choosing between two fixed viewpoints.

This work is motivated by the same type of insight that led to the rich corpus of work on active vision where observer motion was used to enhance the performance of classic (or novel) local estimators or even computation of intrinsic images [21]. In his early work, Davidson [22] considered the use of active vision but primarily as a mechanism for identifying single features for stereo fixation. The present work, however, selects broad gaze directions for ego-motion estimation in real-time.

III. Approach

Our approach is broken down into three components: 1) visual odometry and SLAM, 2) texture classification, and 3) gaze control based on the outputs of 1 and 2.

A. Visual odometry and SLAM

Our core localization system (as shown in Fig. 2) is based on the SLAM algorithm ORBSLAM [11] with some modifications that allow the use of a stereo camera to initialize map points. The system (dubbed MORESLAM [3]) is tuned to provide a combination of power efficiency, real-time feedback, and robustness, so as to allow real-time SLAM on a robotic platform. Additionally, MORESLAM allows tight integration of readings from an Inertial Measurement Unit (IMU) if one is available, using them to predict keypoint locations in consecutive frames, thus reducing the matching time and boosting efficiency. In the following description of MORESLAM, references to an IMU are intended to be conditional on its presence.

MORESLAM estimates the pose of the left camera LC in the fixed world frame-of-reference W. The right camera RC

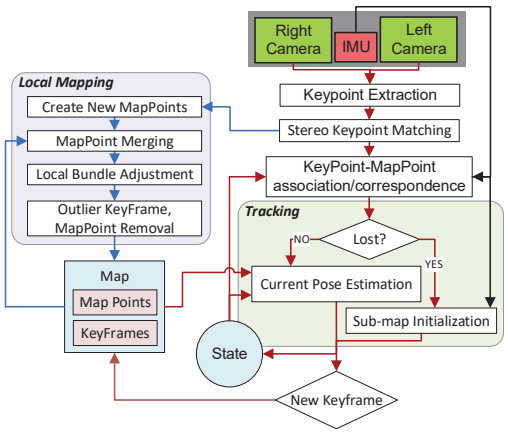

Fig. 2. SLAM pipeline beginning with our camera/IMU sensor and ending with the state estimate.

and IMU I frames are fixed with respect to each other and the left camera, and we assume their relative positions are known in advance (we calibrate our system using the Kalibr software toolbox [23]).

The state vector S estimated by our system is:

$$\mathbf{S} = \begin{bmatrix} {}^{LC}_{W}\mathbf{q} & {}^{LC}\mathbf{p}_W & {}^{W}\mathbf{v}_I & \mathbf{b}_g & \mathbf{b}_a \end{bmatrix} \quad (1)$$

where ${}^{LC}_{W}\mathbf{q}$ is a unit quaternion that expresses the rotation from the fixed world frame to the left camera frame, ${}^{LC}\mathbf{p}_W$ is the position of the world frame in the coordinates of the left camera frame, ${}^{W}\mathbf{v}_I$ is the velocity vector of the IMU in world frame, $\mathbf{b}_g$ is the gyroscope bias, and $\mathbf{b}_a$ is the accelerometer bias affecting the IMU measurements.

Our cameras run at approximately 15Hz while our IMU is sampled at 100Hz. Both are synchronized using a hardware trigger. At each new camera frame, an abstract data-object is created that stores the instantaneous state information on the IMU measurements that have taken place between each frame. Our cameras are calibrated using the pinhole camera model and share the same capture properties, such as exposure time and gain.

The *Map* maintains a sparse graph of keyframes, which are frames that are chosen when sufficient translation has taken place along a trajectory that leads to sufficient baseline for motion stereo while maintaining some overlap with a nearby keyframe to estimate pose using triangulation. Each keyframe stores a copy of the corresponding image frame along with keypoints extracted from the images.

The *Local Mapping* thread is responsible for creating new map points from pairs of keypoints (which are extracted from each image) between frames and merging existing map points. Map points are stored in the *Map* containing their 3D position (and covariance) in the world frame, the associated keypoints, and its ORB descriptor. The thread also performs local bundle adjustment to refine the position estimate.

The *Tracking* thread is responsible for estimating the current pose of each new frame and determining if the system is still tracking enough map points or not (in which case it is "lost"). The pose estimate is also used to determine whether to add a new keyframe to the map based on the amount of overlap between the views of the nearest keyframe and the current estimated pose.

Since the primary focus of this paper is not visual odometry or SLAM, we have not included the details of our system's implementation (these details may be found in [3]). However, we do require a method for expressing how the uncertainty in the position estimate of a map point propagates as it is observed over time, as this directly relates to its stability.

The existing Gaussian estimate of a map point expressed in world coordinates, ${}^{W}\mathbf{f}_{t-1}$, has an uncertainty represented by $\mathbf{\Sigma}_{t-1}$, a 3×3 marginal covariance matrix. This uncertainty is due to the error of the initial depth of the map point, which is due to the uncertainty in pixel location (and limited image resolution). When the map point is observed again in a neighboring keyframe with pose ${}^{LC}_{W}\mathbf{q}$, ${}^{LC}\mathbf{p}_W$ and a pixel location of z, the minimum mean squared error is updated as follows:

$$\begin{aligned} \mathbf{K} &= \mathbf{\Sigma}_{t-1}\mathbf{J}_g^T(\mathbf{J}_g\mathbf{\Sigma}_{t-1}\mathbf{J}_g^T + \mathbf{\Sigma}_z)^{-1} \\ \mathbf{\Sigma}_t &= \mathbf{\Sigma}_{t-1} - \mathbf{K}\mathbf{J}_g\mathbf{\Sigma}_{t-1} \end{aligned} \quad (2)$$

where $\mathbf{g}$ is the function that projects a point from the world frame to a pixel in the camera's image plane, and $\mathbf{J}_g$ is its Jacobian evaluated at the previous mean estimate ${}^{W}\mathbf{f}_{t-1}$. $\mathbf{\Sigma}_z$ is a diagonal matrix used to give higher weights to ORB keypoints detected at nearer scales - which have less uncertain depths - than to those detected at farther scales.

We use this uncertainty in the proceeding steps when learning the "goodness" of a texture class for visual odometry. In principle, this methodology can apply equaly well to any keypoint-based visual odometry and SLAM algorithm that maintains an uncertainty estimate of the locations of its keypoints.

B. Texture-based patch classification

One requirement of this work is that it achieves a level of computational efficiency that allows our gaze-direction system to be used in real time as a component of a larger system on a small UAV or other robot. Classifying image patches provides a low-dimensional abstraction of the texture they contain, which enables us to use the low-cost scoring process described in III-C. However, the patch classification itself must also be computationally efficient.

Our patch classifier uses a histogram of LBP descriptors as its input feature. LBP descriptors constitute a highly computationally optimized texture operator and have been widely used in texture analysis. They capture aspects of both contrast and local appearance while providing invariance to global illumination changes. The details of the LBP computation can be found elsewhere [4], but the essential computation is a multi-scale comparison between a center pixel and its circularly-distributed neighbors. Given a center pixel c, and P equally spaced points about a circle with radius R surrounding the center pixel, the basic LBP descriptor is

defined as

$$LBP_{P,R} = \sum_{p=0}^{P-1} \mathbb{1}_{\{g_p - g_c \geq 0\}} 2^p, \tag{3}$$

where g_c and g_p is the grayscale intensity of the center pixel and the pixel at each surrounding point respectively, and $\mathbb{1}_{\{\cdot\}}$ denotes the indicator function. In our work, we use a variant of the basic LBP that guarantees rotational invariance [3]. We experimented with parameters over the range $R = \{1, 2, 3\}, P = \{8, 16, 24\}$ and found that $R = 2$ and $P = 16$ yielded patch classifications that enabled beneficial active gaze control.

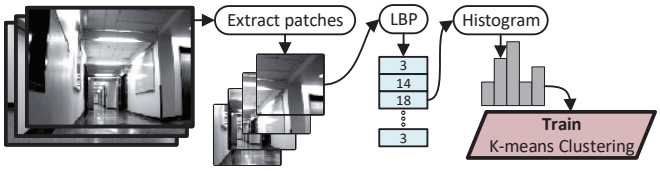

Fig. 3. Pipeline of extracting LBP descriptors, building a histogram and training a K-means classifier.

For each 40×40-pixel patch in an image, a histogram of LBP descriptors is computed using the LBP response for each pixel in the patch, forming a descriptor of the patch. As depicted in Fig. 3, using the descriptors of patches extracted from a small corpus of training images, k-means clustering is performed to find k patch cluster centers. Then online, we compute descriptors of visible patches and classify each one according to the nearest cluster center. The parameter k determines the number of classes we can distinguish. A larger number of classes can provide better results but requires a larger amount of training data. We set the number of classes based on the constraints of each experiment.

The combination of efficient LBP features with nearest-cluster-center classification results in a very fast patch classifier and has negligible effect on the SLAM algorithm running on the same CPU. The patch classes are used in the following step to determine a "good" gaze direction.

C. Gaze control

The pipeline used to determine the desired gaze direction is shown in Fig. 4, which continually updates a score for each of the texture classes learned by the classifier. New images from the left camera in the stereo pair are segmented into 40x40-pixel patches as they arrive (for simplicity, the right camera is ignored). The classifier is run on each of the patches to assign it to a class. Independently, a look-up table is maintained that associates each texture class with a "goodness" score. When a map point is observed in an image, the location of that point in the image corresponds to a specific patch, which in turn is associated with a specific texture class. It is this class's score that is updated each time the map point is observed.

The score update begins by computing a quality score for the map point, which is:

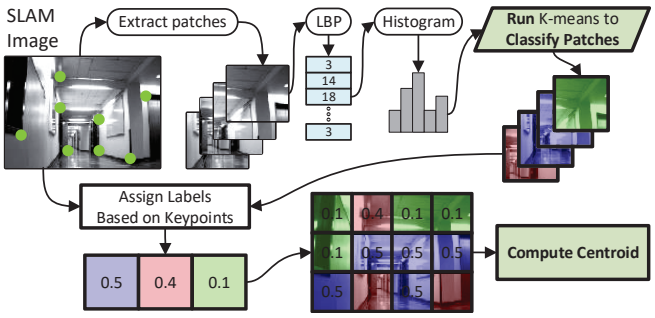

Fig. 4. Pipeline of assigning scores to texture classes and computing a weighted centroid used to direct the gaze.

$$s_{mp} = N_{obs}/\sigma_z \tag{4}$$

where σ_z is the bottom component of the diagonal of $\mathbf{\Sigma}_t$ as defined in Eq. 2, which represents the depth variance of map point, and N_{obs} is the number of keyframes that have observed the map point (which indicate the stability of the point).

Using this map point score, the quality score for the texture class is updated:

$$s_{class} = \frac{\mu_{s_{mp}} * N_{mp}}{\sigma_{s_{mp}}} \tag{5}$$

where $\mu_{s_{mp}}$ is the average score of all the map points observed in patches of this class, $\sigma_{s_{mp}}$ is the standard deviation of the same quantity, and N_{mp} is the number of map points observed in patches of this class.

To determine where the gaze should be directed, the aforementioned image patches are each assigned a weight equal to the score of their class, and the centroid of these weighted patches is then computed. We take this centroid to be the highest-quality gaze direction. Using the camera's intrinsic parameters, the location of the centroid in image coordinates is converted to an angle away from the center of the camera's field of view as follows:

$$\theta_x = \arctan(c_x * \tan(FOV_x/2)) \tag{6}$$

$$\theta_y = \arctan(c_y * \tan(FOV_y/2)) \tag{7}$$

where c_x, c_y are the coordinates of the centroid, each ranging from -1 at the left/top edge of the image to 1 at the right/bottom edge, and FOV_x, FOV_y are the horizontal and vertical fields-of-view in radians of the camera that took the image. This angle is added to the current gaze direction to obtain a new desired gaze direction. The new gaze direction will result in the camera pointing towards the area of its current view that has the highest "quality" for detecting stable features. New desired gaze directions are computed continuously in real-time, constantly updating the camera's gaze direction as it receives new images.

Fig. 5 shows a visualization of the patch-class assignment of an image and the centroid computed from the weighted patches.

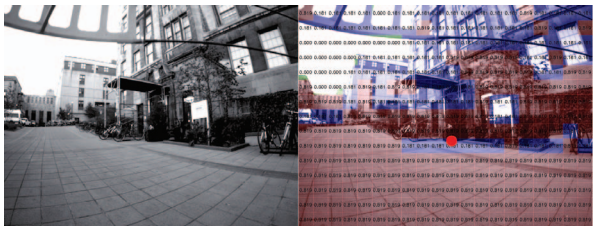

Fig. 5. A representative visualization of the gaze selector, along with the raw image. Each image patch is assigned to a class, represented here by a color. The score of that class is overlayed on the patch. The centroid is shown as a red circle.

Fig. 6. Overhead view of one of our simulated environments. The blue track shows the path taken by the robot through the environment (it is not visible in the simulator).

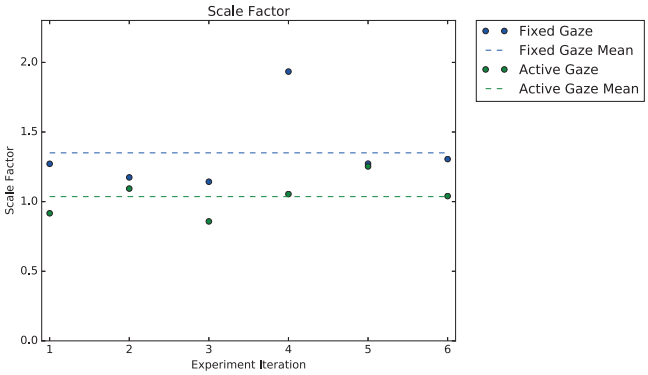

Fig. 7. Scale errors in the position estimation as a ratio of the estimated trajectory size to the real trajectory size for the six fixed-gaze trials and the six gaze-control trials in simulation. The means of each set of six experiments are shown as horizontal lines.

Our methodology separates the training of the classifier (which is performed offline) from the online quality scoring of the texture classes. Before running the gaze control online, a data-gathering and training phase is carried out where a small sample of images (typically 30-50) is acquired from an environment similar to that where the system is intended to run. The k-means classifier is trained on this data and saved for use while running the active gaze control.

The quality score for each texture class is updated online while the robot is running. In addition, we have the ability to save the class-to-score look-up table at any given time. Using a saved look-up table allows us to initialize the table with non-zero scores that provide an initial estimate of a ”good” gaze direction. The loaded look-up table is then continually updated in the same manner described above. This approach is advantageous when running the system on a highly sensitive robotic platform, such as a UAV, where stability is important for safety reasons.

IV. EXPERIMENTAL EVALUATION

Our experimental validation is based on three classes of assessment: (1) simulation studies using a range of scenarios to validate the nature of our approach and the associated performance tradeoffs in a perfectly controlled and repeatable environment, (2) gaze control on fixed outdoor trajectories where the robot’s path is strictly constrained - both to establish reliable ground truth and to carefully measure the impact of gaze shift alone (since in free flight, interactions between gaze shifting and the vagaries of unavoidable trajectory variations make a quantitative assessment unreliable), and (3) an open environment, closed-loop control flight illustrating the feasibility of the method. Note that a common evaluation method for visual SLAM systems is to compare their performance on existing SLAM benchmark datasets such as KITTI ([24]), which consist of pre-recorded visual trajectories. However, since our technique involves active control, it is not suitable for evaluation on pre-recorded benchmarks.

A. Simulated experiments

A series of experiments were carried out in a simulated environment using the Gazebo simulator platform. In these experiments, a simulated robot equipped with a stereo camera pair mounted on a pan-tilt unit was made to follow a fixed trajectory of waypoints forming a ”figure-eight” pattern through an artificial 3D environment consisting of textured models of real-world-like objects, such as houses and traffic cones. The pan-tilt unit was situated at a height of two meters (in simulated units) above the ground plane, and the cameras were attached one meter in front of the pan-tilt unit’s rotational center with stereo baseline of 0.4 meters. Objects in the environment ranged in height from less than one meter to 10 meters. The environment and the robot’s path through it are shown in Fig. 6.

For this environment, the patch classifier was trained with k = 4 classes. Six trials were conducted with the stereo cameras always facing in the direction of the robot’s motion (fixed gaze), and six trials were conducted in which the pan-tilt unit was controlled by commands from our gaze-control system as the robot moved through the environment. Our SLAM system was run concurrently in real-time during all trials. The SLAM system’s estimated trajectory over each trial was then compared with the true trajectory.

For each trial, we first computed a scale factor from the size of the actual traversal to the estimated size of the traversal. The scale factor was based on the maximum and minimum estimated positions of the robot in each of the X and Y dimensions, compared to the actual scale of the trajectory on each axis. The scale factors are depicted in Fig. 7. In the fixed-gaze trials, the scale was consistently overestimated by a factor of 1.35 on average. By contrast, under gaze control the SLAM system’s scale estimates were much more accurate, averaging 1.05.

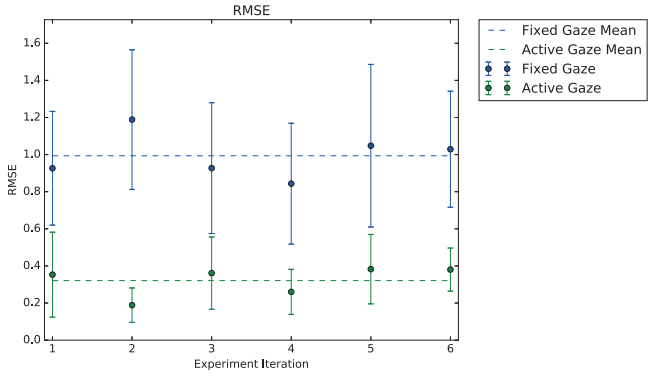

Fig. 8. RMSE errors of the estimated robot pose of the six fixed-gaze trials and gaze-control trials all in the simulation environment. The mean of each set of six experiments are shown as horizontal lines. Since it is a virtual environment, error is measured in virtual, but consistent non-physical units.

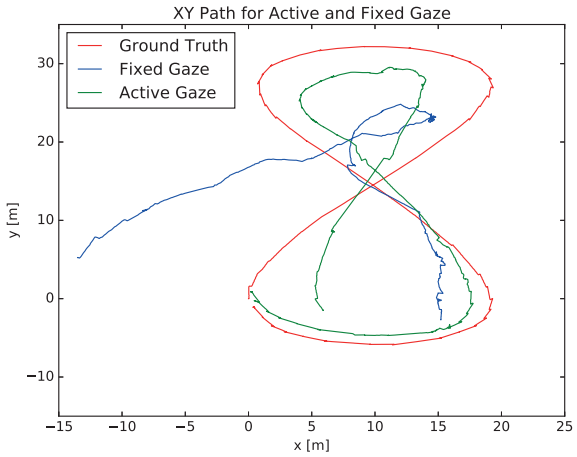

Fig. 9. The ground truth trajectory and the estimated trajectories from both fixed gaze and gaze-control trajectories in the 4th pair of trials.

To compute the error of each estimated trajectory, the trajectory was first scaled by the inverse of the scale factor computed for it in the previous step, so as to separate our estimate of the scale error from that of the position error. The root-mean-squared-error (RMSE) was computed between the estimated and true poses over the whole trajectory along with the standard deviation of the positional errors. The results for the six trials are shown in Fig. 8. As the figure shows, the average RMSE over the fixed-gaze trials was three times that of the directed gaze trials. Gaze control hugely improved the accuracy of our SLAM technique, reducing scale error on average by a factor of seven and positional error on average by a factor of three.

These findings are also illustrated in Fig. 9, which shows the ground truth trajectory and the estimated trajectories from the fourth fixed-gaze and gaze-control trials. We chose to highlight the fourth trial as it was the trial where the smallest RMSE of all the fixed-gaze trials was achieved. Drift is evident in both estimated trajectories, but the gaze-control estimate preserved the figure-eight pattern closely. In contrast, the fixed-gaze estimate's drift was extreme, with its ending position being far from its initial position.

B. UAV platform

All real-world experiments were conducted on a custom-built 420-mm quadrotor platform weighing 2.5 kg (including battery), based on an Astec Pelican frame. The camera-IMU sensor consisted of a VectorNav VN100 IMU and two uEye UI-3251-LE-C-HQ cameras. The quadrotor was also equipped with an Intel NUC Kit NUC5i7RYH mini-PC, which received input from the cameras and IMU and ran the SLAM and directed-gaze algorithms described in this paper. An on-board autopilot attempted to satisfy the goal gaze from the gaze control system and position goals listed in a pre-written list of waypoints, using the state estimate from the SLAM system as the measure of its current pose.

In practice, because our robot platform was limited to a roughly planar motion and the cameras were fixed relative to it, the goal gaze in these experiments was only provided as a desired change in yaw angle in the horizontal plane. However, it can easily apply to robots with greater freedom of gaze direction.

We ensured that the initial map points created were computed using stereo to achieve an initially accurate scale. However, in the experiments conducted on our UAV, we limited the number of stereo matches used by our SLAM system while in flight to reduce the computational overhead. We trained the patch classifier with k = 3 classes for all experiments that were run on the UAV.

C. Constrained path experiments

We performed experiments using a constrained repeatable "flight" path in which the UAV was not permitted to actually perform free flight, but instead was attached to a moving platform that was 81.5 cm in height. The system ran live in real-time on the UAV while the platform was driven along a pre-determined route: a right-angle path around two sides of the James Administration building at McGill University. We performed a series of traversals of the same path with five fixed-gaze directions, ranging from a gaze perpendicular to the left of the robot's heading, through a straight-ahead gaze, to a gaze perpendicular to the right: $-90°$, $-45°$, $0°$, $45°$, $90°$. Finally, we conducted a traversal using directed gaze. In this traversal, the UAV began with a straight-ahead gaze (yaw $= 0°$). The directed gaze system periodically displayed a desired change in yaw, and the UAV's facing direction on the moving platform was adjusted to point it in the desired direction to within two degrees (precision being limited by the operator's dexterity). This step was performed approximately once per meter. If the robot's requested yaw change at a one-meter interval was less than one degree, the yaw was not adjusted.

It must be noted that for each fixed-gaze traversal, the SLAM system was initialized in the fixed gaze direction, and features at different distances were visible in the different directions, potentially leading to differences in the initial

scale that would affect the accuracy of the rest of the estimated trajectory. The distances between the SLAM system's estimated end-points and the actual end-point are give in Table I. Of all the traversals, the directed-gaze traversal (the first row in Table I) had the smallest metric error between the estimated and actual end-points, although the 90°-yaw traversal was close.

TABLE I

ERROR BETWEEN ESTIMATED AND TRUE END POSITIONS FOR THE JAMES PATH.

traversal	final position estimate error (m)
dynamic directed yaw	**10.2551**
0 degrees fixed yaw	27.6299
-45 degrees fixed yaw	17.8358
-90 degrees fixed yaw	19.6973
+45 degrees fixed yaw	15.7366
+90 degrees fixed yaw	11.4023

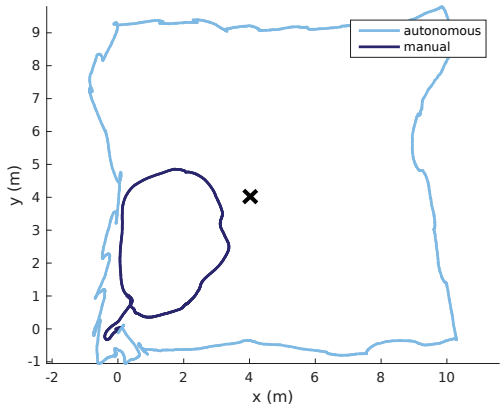

Fig. 10. SLAM system's position estimates for the park flights. The black X shows the approximate location of the sculpture, +/- 50 cm. It is clear that the autonomous active-gaze control flight, provided a much better estimate of the trajectory travelled.

D. Free-flight experiments

To validate the feasibility of our approach and demonstrate that our simulation and constrained experiments lead to practically exploitable methods, we deployed our UAV with the gaze control system integrated with the flight controller in a number of indoor and outdoor environments. Precise positional ground truth was not available in these experiments; their purpose was to validate our system and show that it can be effectively used in real-world environments on a resource-constrained robotic platform.

In these scenarios, closed-loop control was used during free-flight to modulate the gaze direction as above while following a set of pre-programmed waypoints. We repeatedly found that the UAV's closed-loop control system was able to maintain stable control of the vehicle while simultaneously follow waypoints and perform gaze adjustments.

One prototypical scenario was a series of flight tests in an outdoor, park-like environment. This environment contained many challenging objects for SLAM systems such as trees and grass, which are rich in texture but prone to movement, thus making visual features extracted from them difficult to track. Two of these flights are illustrated in Fig. 10. One flight was performed under closed-loop, autonomous waypoint-following with gaze control, which kept the gaze mostly constant and counteracting the autopilot's default goal of facing in the direction of travel. The other flight was performed under manual control along approximately the same route with the same takeoff and landing positions within 30 cm but with the gaze fixed towards the direction of travel. For both flights, the UAV took off, made a square loop about 12 m wide and centered at a large sculpture (approximately 3 m tall and 1 m square marked by the black 'x' in Fig. 10), and returned to a fixed landing point. We used the relative position of the sculpture and the UAV as a form of qualitative ground truth.

During the autonomous flight, the system's estimate of its trajectory was a square loop approximately centered on the sculpture, which qualitatively matched the actual traversal. By comparison, during the manual flight, the SLAM system's scale estimate without gaze control drifted substantially. As shown in Fig. 10, its estimate did not go around the sculpture at all, when in fact it made a wide loop around that object.

V. CONCLUSIONS

In this paper, we have described an approach to using active gaze control to improve the robustness and accuracy of real time vision-based SLAM and navigation. We demonstrate that gaze selection has quantitative and qualitative advantages when used with an autonomous air vehicle. The importance of gaze control has been long recognized in human vision and stereo, but to the best of our knowledge, it has not been exploited in this manner for UAV-based SLAM.

While the emphasis of this paper is on gaze selection and its impact, it also demonstrates the effectiveness of our fused sensing SLAM system, which we have dubbed *MORESLAM*. By combining monocular flow-based SLAM, stereo depth estimation, and an IMU, MORESLAM combines the advantages of each of these subsystems, while still being fast and efficient enough to run in real-time on a UAV alongside our active gaze control module.

While MORESLAM takes advantage of the stereo cameras used in our experiments, the gaze control module only makes use of one of the two cameras. If texture information from both cameras in the stereo pair could be fused, it might provide a better basis for gaze selection. There are likely a number of sensible ways to use information from both cameras for gaze control, which we plan to exploit in future work.

Another natural extension of this work would be to replace the texture classification and patch-weighting steps with an end-to-end online learning process that would learn to produce gaze commands directly from the input images, perhaps using deep reinforcement learning methods. The map-point-quality metric (Eq. 4) for observed points could serve as a basis for a delayed reward signal in such a system.

ACKNOWLEDGMENTS

We would like to thank R. Mur-Artal, J. M. Montiel, and J. Tardos, for making ORB-SLAM1 and 2 publicly available.

REFERENCES

[1] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[2] T. Y. Tian, C. Tomasi, and D. J. Heeger, "Comparison of approaches to egomotion computation," in *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*. IEEE, 1996, pp. 315–320.

[3] T. Manderson, F. Shkurti, and G. Dudek, "Texture-aware SLAM using stereo imagery and inertial information," in *Computer and Robot Vision*. IEEE, 2016.

[4] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *PAMI*, vol. 24, no. 7, pp. 971–987.

[5] ——, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[6] H. von Helmholtz, "Treatise on physiological optics vol. iii," 1867.

[7] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 1403–1410.

[8] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 1–10, Nov. 2007.

[9] A. Harmat, I. Sharf, and M. Trentini, "Parallel tracking and mapping with multiple cameras on an unmanned aerial vehicle," *Intelligent Robotics and Applications*, pp. 421–432, 2012.

[10] C. Forster, M. Pizzoli, and D. Scaramuzza, in *Proc. IEEE Intl. Conf. on Robotics ...*, 2014.

[11] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a Versatile and Accurate Monocular SLAM System," p. 15, Feb. 2015.

[12] S. Shen, N. Michael, and V. Kumar, "Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2015, pp. 5303–5310.

[13] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visualinertial odometry using nonlinear optimization," *IJRR*, vol. 34, no. 3, pp. 314–334, 2015.

[14] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Robotics Science and Systems (RSS)*, June 2015.

[15] S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery, "Augmenting inertial navigation with image-based motion estimation," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 4. IEEE, 2002, pp. 4326–4333.

[16] D. G. Kottas, J. A. Hesch, S. L. Bowman, and S. I. Roumeliotis, "On the consistency of vision-aided inertial navigation," in *Experimental Robotics*. Springer, 2013, pp. 303–317.

[17] M. Li and A. I. Mourikis, "3-d motion estimation and online temporal calibration for camera-imu systems," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5709–5716.

[18] S. Soatto, P. Perona, R. Frezza, and G. Picci, "Recursive motion and structure estimation with complete error characterization," in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*. IEEE, 1993, pp. 428–433.

[19] V. Peretroukhin, J. Kelly, and T. D. Barfoot, "Optimizing camera perspective for stereo visual odometry," in *Computer and Robot Vision (CRV), 2014 Canadian Conference on*, May 2014, pp. 1–7.

[20] K. Wu, T. Do, L. CarrilloArce, and S. Roumeliotis, "On the VINS Resource Allocation Problem for a Dual Camera, Small Size Quadrotor," in *Proc. 2016 International Symposium on Experimental Robotics (ISER)*. Springer, October 2017, p. to appear.

[21] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *International journal of computer vision*, vol. 1, no. 4, pp. 333–356, 1988.

[22] A. J. Davison and D. W. Murray, "Simultaneous localization and map-building using active vision," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 865–880, 2002.

[23] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 1280–1286.

[24] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.