Qualitative Vision-Based Path Following

Zhichao Chen and Stanley T. Birchfield, *Senior Member, IEEE*

***Abstract*—We present a simple approach for vision-based path following for a mobile robot. Based upon a novel concept called the *funnel lane*, the coordinates of feature points during the replay phase are compared with those obtained during the teaching phase in order to determine the turning direction. Increased robustness is achieved by coupling the feature coordinates with odometry information. The system requires a single off-the-shelf, forward-looking camera with no calibration (either external or internal, including lens distortion). Implicit calibration of the system is needed only in the form of a single controller gain. The algorithm is qualitative in nature, requiring no map of the environment, no image Jacobian, no homography, no fundamental matrix, and no assumption about a flat ground plane. Experimental results demonstrate the capability of real-time autonomous navigation in both indoor and outdoor environments, on flat, slanted, and rough terrain with dynamic occluding objects for distances of hundreds of meters. We also demonstrate that the same approach works with wide-angle and omnidirectional cameras with only slight modification.**

***Index Terms*—feature tracking, mobile robot navigation, vision-based navigation, control**

I. Introduction

Route-based knowledge, in which the spatial layout of an environment is recorded from the perspective of a ground-level observer, is an important component of human and animal navigation systems [31]. In this representation, navigating from one location to another involves comparing current visual inputs with a sequence of views captured along the path in a previous instance. Applications that would benefit from such a path-following capability include courier and delivery robots [4], robotic tour guides [32], or reconnaissance robots following a scout [7].

One approach to path following is visual servoing, in which the robot is controlled to align the current image with a reference image, both taken by an onboard camera [14]. Such an approach generally employs a Jacobian to relate the coordinates of world points to their projected image coordinates [5], a homography or fundamental matrix to relate the coordinates between images [29], [20], [27], [36], or bundle adjustment to minimize the reprojection error over multiple image frames [28]. As a result, the camera usually must be calibrated [5], [27], [28], [36], and even uncalibrated systems require lens distortion to be removed. Alternative vision-based algorithms make strong assumptions about the environment or the sensor, such as a flat ground plane [5], [20], [12], [29], a man-made environment in which vertical straight lines are present [16], [12], [29], [34], or an omnidirectional camera [10], [35], [18], [34].

To overcome these limitations, we consider the problem from a novel viewpoint in which there is no equation relating image coordinates to world coordinates. Such a direct approach is motivated by the observation that the problem is vastly overdetermined, with tens of thousands of image pixels available to determine a single turning command output. We present a simple algorithm that uses a single, off-the-shelf camera attached to the front of the robot. The technique follows the teach-replay approach [5] in which the robot is manually led through the path once during a teaching phase and then follows the path autonomously during the replay phase. Without any camera calibration (even calibration for lens distortion), the robot is able to follow the path by making only qualitative comparisons between the feature coordinates in the two phases. All that is needed is a single controller gain parameter to convert pixel coordinates to turning angles. We demonstrate the technique on several indoor and outdoor experiments, showing its robustness with respect to slanted surfaces, changing lighting conditions, and dynamic occluding objects. This paper extends the applicability and improves upon the robustness of our earlier work [6] by incorporating odometry information and correcting for camera roll. We also demonstrate the ability of the technique to work with wide-angle and omnidirectional cameras, with only slight modification in the latter case to ignore the bottom half of the image which views the scene behind the robot.

The proposed approach falls within the category of mapless algorithms [8]. As such, it is closely related to the view-sequenced route representation (VSRR) of Matsumoto et al. [21], [22], [15] in which the turning angle is computed by cross-correlating images acquired during the replay phase with those captured during training. However, VSRR requires large amounts of memory to store the views and is sensitive to occlusions by dynamic objects. Along with a homography-based extension using vertical lines [29], it has only been demonstrated for short sequences on flat terrains. An alternate mapless approach is to learn the mapping from images to turning commands based on their classification [37], [1]. While this method can successfully follow a specific pattern such as a road or hallway, it will have difficulty generalizing to environments in which the images cannot be categorized into a small number of classes known at training time. Another approach that has received considerable attention [10], [38], [39], [33], [17], [35], [13] is to store an example image with each specific location of interest. At run time, the image database is searched to find the image that most closely resembles the current one (or, alternatively, the current image is projected onto a manifold learned from the database [25], [18]). Such approaches require extensive training and have difficulty providing sufficient spatial resolution to determine actual turning commands in large environments. Similarly, sensory-motor learning has been used used to map visual inputs to turning commands, but the resulting algorithms have been too computationally demanding for real-time performance [11]. Other researchers have developed mapless

algorithms for low-level functionality like corridor following or obstacle avoidance [26], [30], [2], [19], [23], [24], but these techniques are not applicable to following a specific arbitrary path.

II. Qualitative mapping from feature coordinates to turning direction

Consider a mobile robot equipped with a camera whose optical axis is parallel to the heading direction of the robot. Suppose we wish to move the robot from location $C = (x_C, y_C, \theta_C)$ to a previously encountered location $D = (x_D, y_D, \theta_D)$, where (x_i, y_i) and θ_i are the position and orientation, respectively, in the xy plane, $i \in \{C, D\}$. The robot has access to a current image I_C, taken at C, and a destination image I_D, taken previously at the destination D.

We start with a simple observation. Suppose the robot views a fixed landmark in both images yielding image feature coordinates of u^C and u^D, as shown in Figure 1. The features are computed with respect to a coordinate system centered at the principal point (the intersection of the optical axis and the image plane), so that positive coordinates are on the right side of the image while negative coordinates are on the left side. If the robot moves toward the destination in a straight line with the same heading direction as that of the destination (i.e., $\theta_C = \theta_D$), then the point u^C will move away from the principal point toward u^D, reaching u^D when the robot reaches D. This observation is made more precise in the following theorem.

Theorem 1: Let a mobile robot move in a straight line toward location D on a flat surface. Let u^j be the horizontal image coordinate, relative to the principal point, of a monotonic projection at location j of a fixed landmark. For any location C along the line such that $\theta_C = \theta_D$, $|u^C| < |u^D|$ and $\text{sign}(u^C) = \text{sign}(u^D)$.

The theorem can be easily proved by geometry. Note that the image projection function is only required to be monotonic (i.e., perspective projection is not necessary), so the result applies equally to a camera with radial lens distortion. The primary assumption is that the optical axis of the camera passes through the axis of rotation of the robot. Other assumptions include the alignment of the optical axis with the robot heading direction, zero roll and tilt angles of the camera with respect to the robot, and a flat ground plane. In practice, misalignment is not an issue because the camera alignment can be learned automatically by estimating the focus of expansion as the robot drives forward. Similarly, rough terrain is easily handled by measuring image rotation to compensate for a non-zero roll angle of the robot and by recognizing that a non-zero tilt angle has a negligible effect on the horizontal feature coordinates.

A. The funnel lane

According to the preceding theorem, if the robot is on the path toward the destination with the same heading direction, then two constraints are satisfied. Conversely, as shown in Figure 2, if the constraints are satisfied then the robot lies within a trapezoidal region (assuming perspective projection) for any given relative robot angle $\alpha = \theta_C - \theta_D$. For $\alpha = 0$

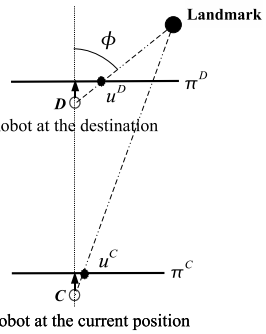

Fig. 1. The robot is at C moving toward the destination D with the same heading direction. The open circle coincides with both the camera focal point and the robot position, the arrow indicates the heading direction, π is the image plane, and ϕ is the angle between the optical axis and the projection ray from the landmark.

the sides of the trapezoid are defined by two lines passing through the landmark, one through D and another that is parallel to the destination direction. These lines are rotated about the landmark by α if the relative angle is nonzero. We call the trapezoidal region the *funnel lane* associated with the landmark, destination, and relative angle. The terminology arises from the analogy of pouring liquid into a funnel: The liquid moves in a straight line until it hits the sides of the funnel, which cause it to bounce back and forth until it eventually reaches the spout. In a similar manner, the sides of the trapezoid act as bumpers, guiding the robot toward the goal. The notion of the funnel and the funnel lane are captured in the following definitions.

Definition 1: The *funnel* of a fixed landmark λ and a robot location D is the set of locations $\mathcal{F}_{\lambda,D}$ such that, for each $C \in \mathcal{F}_{\lambda,D}$, the two funnel constraints are satisfied:

$$\begin{aligned} |u^C| &< |u^D| && \text{(Constraint 1)} \\ \text{sign}(u^C) &= \text{sign}(u^D) && \text{(Constraint 2)} \end{aligned}$$

where u^C and u^D are the coordinates of the image projection of λ at the locations C and D, respectively.

Definition 2: The *funnel lane* of a fixed landmark λ, a robot location D, and a relative angle α is the set of locations $\mathcal{F}_{\lambda,D,\alpha} \subset \mathcal{F}_{\lambda,D}$ such that $\theta_C - \theta_D = \alpha$ for each $C \in \mathcal{F}_{\lambda,D,\alpha}$.

Multiple features yield multiple funnel lanes, the intersection of which is the set of locations for which both constraints are satisfied for all the features. This intersection, which we call the *combined funnel lane*, is depicted in Figure 2. Notice the importance of having features on both sides of the image in order to narrowly constrain the path of the robot, thus achieving more robust and accurate results. Features can be at any depth, and there need not be any relationship between the depths of the various features as long as they remain visible.

B. Qualitative control algorithm

The funnel constraints lead to a simple control algorithm, illustrated in Figure 3. The robot continually moves forward, turning to the right whenever Constraint 1 is violated and to the left whenever Constraint 2 is violated, given a feature on

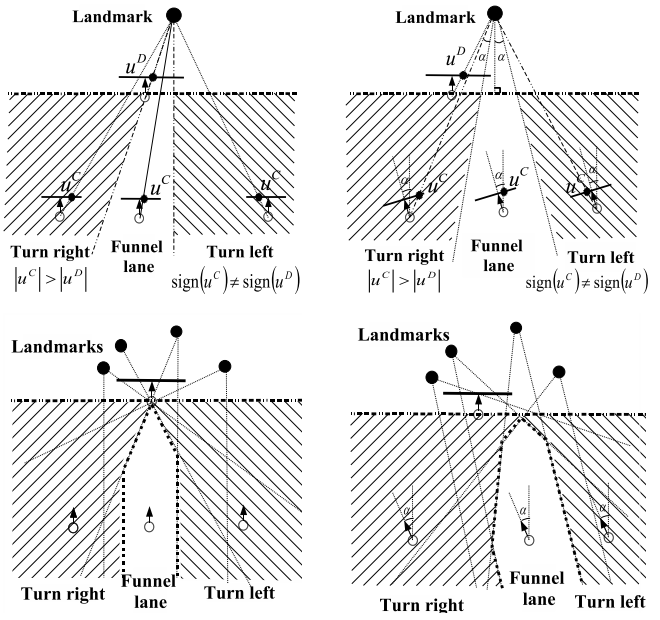

Fig. 2. TOP: The funnel lane created by the two constraints, shown when the robot is facing the correct direction (left) and when it has turned by an angle α (right). BOTTOM: The combined funnel lane created by multiple feature points, shown when the robot is facing the correct direction (left) and when it has turned by an angle α (right).

the right side of the image ($u^D > 0$). If the feature is on the left side ($u^D < 0$), then the directions are reversed.

For each feature i, a desired heading is obtained by

$$\theta_d^{(i)} = \begin{cases} \gamma \min\{u^C, \phi(u^C, u^D)\} & \text{if } u^C > 0 \text{ and } u^C > u^D \\ \gamma \max\{u^C, \phi(u^C, u^D)\} & \text{if } u^C < 0 \text{ and } u^C < u^D \\ 0 & \text{otherwise} \end{cases}$$

where $\phi(u^C, u^D) = \frac{1}{\sqrt{2}}\left(u^C - u^D\right)$ is the signed distance to the line $u^C = u^D$. Here we approximate the conversion of pixels to radians with a constant gain γ.

At any given time, the desired heading of the robot is given by

$$\theta_d = \eta \frac{1}{N} \sum_{i=1}^{N} \theta_d^{(i)} + (1 - \eta)\theta_o, \tag{1}$$

where N is the total number of feature points, θ_o is the desired heading obtained by sampling a third-order polynomial that is fit to the initial and destination odometry measurements of the segment in the teaching phase, and the factor $0 \leq \eta \leq 1$ determines the relative importance of visual measurements versus odometry measurements. We set $\eta = 0.5$ in our system.

III. TEACH-AND-REPLAY NAVIGATION

The navigation system involves two phases. In the teaching phase, an operator manually moves the robot along a desired path to gather training data. The path is divided into a number of non-overlapping segments. Within each segment, feature points are automatically detected in the first image and tracked in subsequent images. When the percentage of features that have been successfully tracked falls below 50% of the original features in the segment, a new segment is declared. For each feature that is successfully tracked throughout a segment, its

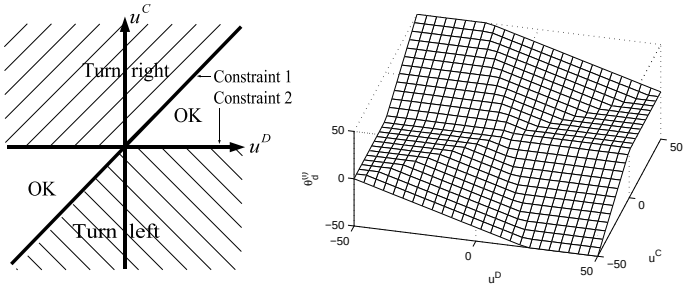

Fig. 3. Qualitative control decision space. The horizontal coordinates of the feature point in the current and destination images (u^C and u^D, respectively) are compared to determine whether to turn the robot to the right, to the left, or not at all. LEFT: Top-down view of decision space. RIGHT: 3D view of decision space, showing the desired angle $\theta_d^{(i)}$ versus u^C and u^D.

graylevel intensity pattern and x-coordinate in the first and last images of the segment are stored in a database for use in the replay phase. We also store the length of each segment and the change of heading direction of the robot in each segment by odometry, which are used in determining the desired heading and the segment transitions.

In the replay phase, the robot automatically proceeds sequentially through the segments starting from approximately the same initial location as that of the teaching phase. At the beginning of each segment, correspondence is established between feature points in the current image and those of the first teaching image of the segment. Then, as the feature points are tracked in the incoming images, their coordinates are compared with those of the *milestone image* (i.e., the last teaching image of the segment) in order to determine the turning direction for the robot. Prior to comparison, feature coordinates are warped to compensate for a non-zero roll angle about the optical axis by applying the RANSAC algorithm [9] to pairs of random features. This compensation removes the undesirable in-plane image rotation that occurs due to unpaved, rough terrain. Note that this is the only place in the algorithm where the y-coordinates of the features are used.

A crucial component of the technique is determining when to transition to a new segment. To solve this problem, we continually monitor the probability that the robot at time t is at the end of the current segment:

$$\delta(t) = \underbrace{\exp\left\{-\frac{\epsilon_f^2(t)}{2\sigma_f^2}\right\}}_{\text{feature}} \underbrace{\exp\left\{-\frac{\epsilon_d^2(t)}{2\sigma_d^2}\right\}}_{\text{distance}} \underbrace{\exp\left\{-\frac{\epsilon_h^2(t)}{2\sigma_h^2}\right\}}_{\text{heading}}, \tag{2}$$

assuming that the feature, distance, and heading measurements are independent. In this equation $\epsilon_f(t)$ is the mean squared error of the feature coordinates between the current and milestone images; $\epsilon_d(t)$ is the difference between the distance traveled in the current segment and the corresponding segment in the teaching phase, calculated by odometry; and $\epsilon_h(t)$ is the difference between the current heading and the heading at the end of the teaching segment. These errors are normalized by values computed automatically by the system: σ_f is the mean squared error of the feature points at the beginning of the segment; σ_d is the length of the segment calculated

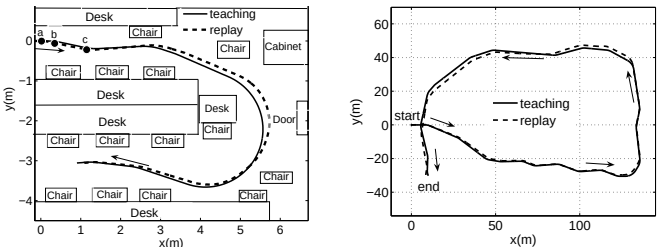

Fig. 4. The teaching and replay paths of the robot in an indoor environment (left), and an outdoor environment (right).

by odometry in the teaching phase; and σ_h is the maximum variation in heading encountered during the teaching segment.

Two values are actually computed: $\delta(t)$ using the current milestone image and $\delta^-(t)$ using the previous milestone image. If $\delta(t-1)-\delta(t) > \tau$ and $\delta^-(t-1)-\delta^-(t) > \tau$, where $\tau = 0.05$, then the system advances to using the next milestone image. The rationale is that both $\delta(t)$ and $\delta^-(t)$ increase as the robot approaches the end of the segment then decrease afterward. Therefore, when both values have decreased by a significant amount, the end has been reached. We have found that using both values yields improved results compared with using a single value. To reduce the effects of noise, both signals are first smoothed by a low-pass nonlinear filter.

IV. Experimental results

The qualitative algorithm was implemented in Visual C++ on a Dell Inspiron 700m laptop (1.6 GHz) controlling an ActivMedia Pioneer P3-AT mobile robot with an inexpensive Logitech QuickCam Pro 4000 webcam mounted on the front. The 320×240 images were acquired at 30 Hz and processed by the KLT algorithm with the default 7×7 feature window size [3]. In all experiments a maximum of 60 features were detected and tracked throughout each segment. On average 85% of the features survive the initial correspondence in the first image of the segment during replay.

The algorithm was tested in a number of indoor and outdoor environments.[1] Figure 4 shows two typical runs in which the robot successfully navigated between chairs and desks along a 10 m path in our laboratory, as well as along a 380 m loop trajectory in a parking lot of our university campus. The driving speed of the robot was 100 mm/s and the turning speed was 4 degrees per second during both the teaching and replay phases of the indoor experiments. Outdoors, the additional maneuvering room enabled the driving and turning speeds to be increased to 750 mm/s (the maximum driving speed of the robot) and 6 degrees per second, respectively. The error was less than 1 m for two-thirds of the sequence and remained below 2.5 m for the entire sequence.

Figure 5 shows sample images from two experiments demonstrating the robustness of the algorithm. In the first, the robot navigated a slanted ramp in a 40 m run, thus verifying that the algorithm does not require a flat ground plane. In the second, the robot navigated a narrow road for 80 m

[1]Videos of the results can be found in the multimedia attachment or at `http://www.ces.clemson.edu/~stb/research/mobile_robot`

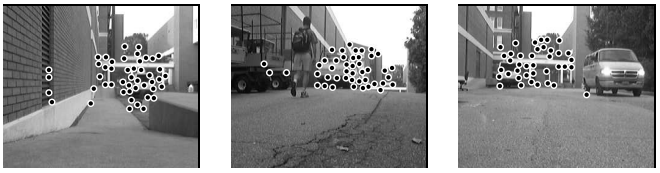

Fig. 5. Sample image frames showing the robot traveling down and up a ramp and past dynamic objects. The circles indicate the features.

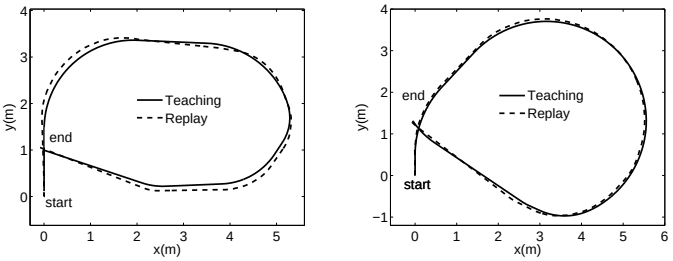

Fig. 6. The approach successfully following a path using a wide-angle camera (left) and an omnidirectional camera (right).

while a pedestrian walked by the robot and later a van drove by it. Because the milestone images change frequently, the algorithm quickly recovered from the loss of features due to the occlusion caused by the dynamic objects.

Similarly, Figure 6 shows the results of the approach using cameras with severe lens distortion. In one experiment we used a wide-angle camera with a 3.5 mm focal length and 110-degree field of view. The other experiment utilized an omnidirectional camera with a 360-degree field of view. For both experiments we used the same parameters as the previous experiments. The only change made to the code was to discard the bottom half of the omnidirectional donut image. This step was necessary because features behind the robot (whether viewed by an omnidirectional or standard camera) move in a way that violates the fundamental assumptions of our approach. In contrast, features in front of the camera obey the funnel constraints sufficiently to be of use in keeping the robot on the path, despite their moving in curved image paths due to the severe lens and catadioptric distortion. The average error of the two experiments was 0.04 m and 0.04 m, respectively, while the maximum error was 0.13 m and 0.09 m.

To further illustrate the lack of calibration, we conducted an outdoor experiment in which the robot navigated the same 50 m path twice. In the first run the robot used the Logitech Quickcam Pro 4000 camera, while in the second run it used an Imaging Source DFK21F04 Firewire camera with an 8.0 mm F1.2 lens. The same camera was used for both teaching and replay. As shown in Figure 7, the algorithm was able to successfully follow the path using either camera, without changing any parameters between runs.

Three additional experiments are shown in Figure 8. In the first, a scout robot was sent along an outdoor path. Another robot, which received the transmitted path information, was then able to follow the same path as the scout. This demonstrates a natural application to swarm robotics, where calibrating dozens or hundreds of cameras would be prohibitive, especially if recalibration is needed whenever the

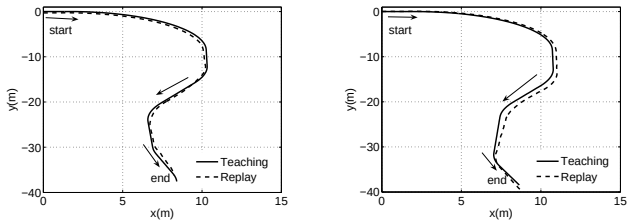

Fig. 7. Teaching and replay paths for the robot using two different uncalibrated cameras, with the same system parameters. LEFT: Logitech QuickCam Pro 4000 USB webcam, RIGHT: Imaging Source DFK 21F04 Firewire camera.

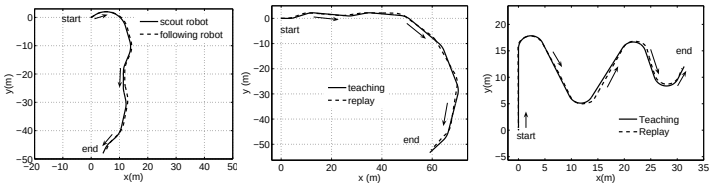

Fig. 8. LEFT: The robot followed a path taken earlier by a scout robot. MIDDLE: A path on rough terrain. RIGHT: A path with sharp turns.

lenses are refocused or the cameras adjusted. The second experiment shows the robot following a path along rough terrain, in which roll and tilt angles up to 5 degrees were encountered. The roll angle compensation described earlier was sufficient to enable the robot to remain on the path. In the third, a path with several sharp turns is demonstrated. This ability is achieved by setting the replay driving speed to be that of the teaching driving speed, which is decreased during a turn.

Additionally, the algorithm was tested in various scenarios to quantitatively measure its accuracy and repeatability. Table I displays the results of the algorithm compared with those of the earlier version [6] which did not use odometry, relied upon a bang-bang control scheme, and did not compensate for the camera roll angle. The algorithms were tested in three environments: a 15 m path in an indoor laboratory environment with rich texture for feature tracking, a 60 m trajectory in an outdoor paved parking lot, and a 40 m path along unpaved terrain. In each case, we conducted ten trials and recorded the final 2D location of the robot for each trial: $\{\mathbf{x}_i\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathbb{R}^2$ and $n = 10$. Accuracy was measured as the RMS Euclidean distance to the final ground truth location: $\sqrt{\frac{1}{n}\sum_{i=1}^{n}||\mathbf{x}_i - \mathbf{x}_{gt}||^2}$. Repeatability was measured as the standard deviation of the final locations: $\sqrt{\frac{1}{n}\sum_{i=1}^{n}||\mathbf{x}_i - \mu||^2}$, where $\mu = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$. While the earlier algorithm works well when the ground is paved and the scenery is rich in texture, the improved algorithm is more robust, achieving maximum errors of only 0.23 m, 1.20 m, and 1.76 m, respectively, compared with 0.45 m, 1.20 m, and 5.68 m for the earlier algorithm.

V. DISCUSSION

Because our system does not explicitly model the geometric world, its geometric accuracy is limited. Therefore, when compared with map-based approaches using calibrated cameras [28], the errors exhibited by the simple control scheme of our algorithm are rather large. Nevertheless, the remarkable flexibility and versatility of the system offer some important advantages over more precise techniques. With our approach, one can literally take an off-the-shelf camera, attach it to the robot, align it approximately in the forward direction, and start the system.

Algorithm	indoor acc. / rep. (m) / (m)	outdoor paved ground acc. / rep. (m) / (m)	outdoor rough terrain acc. / rep. (m) / (m)
vision only [6]	0.30 / 0.18	0.77 / 0.74	3.87 / 1.85
combination (this paper)	**0.14** / **0.08**	**0.60** / **0.55**	**1.47** / **0.66**

TABLE I
COMPARISON OF THE ACCURACY AND REPEATABILITY OF THE ALGORITHM WITH AN EARLIER VERSION, IN THREE DIFFERENT SCENARIOS. THE LOWEST NUMBER IN EACH CASE IS IN BOLD.

The algorithm is not perfect, and there are scenarios in which it will fail. For example, occasionally the algorithm does not properly transition to the next milestone image, in which case the overlap between the current and milestone image can decrease to the point that an insufficient number of features are matched. Also, untextured scenes containing distant trees, bushes, or undecorated indoor hallways sometimes prevent the KLT algorithm from successfully tracking enough features to accurately compute the heading direction. While only a handful of features are necessary for the algorithm to succeed, it is important that features exist on both sides of the image, and that some number of features remain visible throughout the milestone.

Another source of error is due to distant features. Although features near the center of the image produce a narrow funnel lane even when they are far from the camera, distant features near the side of the image produce much larger funnel lanes which are less useful for navigation. Moreover, image parallax is inversely proportional to the distance to a feature. As a result, distant features are primarily useful for correcting the rotation of the robot and are quite incapable of informing the robot about minor translation errors. This problem is compounded by the inherent ambiguity between rotation and translation in the funnel lane itself. Even though this ambiguity has little effect when the robot is near the path, it hinders the ability of the visual information to correctly determine the correct amount of rotation when the robot has deviated significantly. Odometry helps to overcome this limitation, and we have conducted experiments in which the robot consistently returns to the path after deviating by several meters. However, much larger deviations either initially or during replay cannot be handled by our present system. At any rate, it should be noted that odometry drift is not an issue because we only store odometry values local to the segment, not in a global coordinate frame.

VI. CONCLUSION

In this paper we have presented a novel approach to the problem of vision-based mobile robot path following using a single off-the-shelf camera. The robot navigates by performing

a qualitative comparison of feature coordinates across the teaching and replay phases, utilizing the novel concept of a *funnel lane*. Vision information is combined with odometry for increased robustness. The algorithm does not make use of the traditional concepts of Jacobians, homographies, fundamental matrices, or the focus of expansion, and it does not require any camera calibration, including lens calibration. It only requires implicit calibration in the form of a controller gain. Experimental results on both indoor and outdoor scenes demonstrate the effectiveness of the approach on trajectories of hundreds of meters, along with its robustness to effects such as dynamic objects, slanted surfaces, and rough terrain. The versatility of the algorithm in working with wide-angle and omnidirectional cameras with only minor modification has also been shown. Future work should be aimed at incorporating higher-level scene knowledge to enable obstacle avoidance and terrain characterization, as well as connecting multiple teaching paths in a graph-based framework to enable autonomous navigation between arbitrary points.

ACKNOWLEDGMENTS

This work was partially supported through a Ph.D. fellowship from the National Institute for Medical Informatics.

REFERENCES

[1] C. Ackerman and L. Itti. Robot steering with spectral image information. *IEEE Transactions on Robotics*, 21(2):247–251, Apr. 2005.

[2] G. L. Barrows, J. S. Chahl, and M. V. Srinivasan. Biomimetic visual sensing and flight control. In *Bristol Conference on UAV Systems*, 2002.

[3] S. Birchfield, 1997. KLT: An implementation of the Kanade-Lucas-Tomasi feature tracker, http://www.ces.clemson.edu/˜stb/klt/.

[4] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum guide-robot. *Artificial Intelligence*, 114(1-2):3–55, 1999.

[5] D. Burschka and G. Hager. Vision-based control of mobile robots. In *Proceedings of the International Conference on Robotics and Automation*, pages 1707–1713, May 2001.

[6] Z. Chen and S. T. Birchfield. Qualitative vision-based mobile robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2686–2692, May 2006.

[7] S. Crawford, M. Cannon, D. Letourneau, P. Lepage, and F. Michaud. Performance evaluation of sensor combinations on mobile robots for automated platoon control. In *ION GNSS Conference*, pages 706–717, 2004.

[8] G. N. DeSouza and A. C. Kak. Vision for mobile robot navigation: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237–267, Feb. 2002.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[10] P. Gaussier, C. Joulain, S. Zrehen, J. P. Banquet, and A. Revel. Visual navigation in an open environment without map. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 545–550, Sept. 1997.

[11] C. Giovannangeli, P. Gaussier, and G. Désilles. Robust mapless outdoor vision-based navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3293–3300, 2006.

[12] J. J. Guerrero and C. Sagüés. Uncalibrated vision based on lines for robot navigation. *Mechatronics*, 11(6):759–777, 2001.

[13] I. D. Horswill. Polly: A vision-based artificial agent. In *Proceedings of the National Conference on Artificial Intelligence*, pages 824–829, 1993.

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

[15] S. D. Jones, C. S. Andersen, and J. L. Crowley. Appearance based processes for visual navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 551–557, 1997.

[16] A. Kosaka and A. C. Kak. Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties. *CVGIP: Image Understanding*, 56(3):271–329, Nov. 1992.

[17] J. Košecká, L. Zhou, P. Barber, and Z. Duric. Qualitative image based localization in indoors environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3–8, 2003.

[18] B. Kröse, N. Vlassis, R. Bunschoten, and Y. Motomura. A probabilistic model for appearance-based robot localization. *Image and Vision Computing*, 19(6):381–391, 2001.

[19] Y. LeCun, U. Muller, J. Ben, E. Cosatto, and B. Flepp. Off-road obstacle avoidance through end-to-end learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 739–746, 2005.

[20] B. Liang and N. Pears. Visual navigation using planar homographies. In *Proceedings of the International Conference on Robotics and Automation*, volume 1, pages 205–210, 2002.

[21] Y. Matsumoto, M. Inaba, and H. Inoue. Visual navigation using view-sequenced route representation. In *Proceedings of the International Conference on Robotics and Automation*, volume 1, pages 83–88, 1996.

[22] Y. Matsumoto, K. Sakai, M. Inaba, and H. Inoue. View-based approach to robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 545–550, 2000.

[23] J. Michels, A. Saxena, and A. Y. Ng. High-speed obstacle avoidance using monocular vision and reinforcement learning. In *Proceedings of the Twenty-second International Conference on Machine Learning*, pages 593–600, 2005.

[24] V. N. Murali and S. T. Birchfield. Autonomous navigation and mapping using monocular low-resolution grayscale vision. In *Workshop on Visual Localization for Mobile Platforms (in association with CVPR)*, June 2008.

[25] S. K. Nayar, H. Murase, and S. A. Nene. Learning, positioning, and tracking visual appearance. In *Proceedings of the International Conference on Robotics and Automation*, pages 3237–3244, May 1994.

[26] R. C. Nelson and J. Aloimonos. Using flow field divergence for obstacle avoidance towards qualitative vision. In *Proceedings of the International Conference on Computer Vision*, pages 188–196, 1988.

[27] A. Remazeilles and F. Chaumette. Image-based robot navigation from an image memory. *Robotics and Autonomous Systems*, 55(4):345–356, Apr. 2007.

[28] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3):237–260, 2007.

[29] C. Sagüés and J. J. Guerrero. Visual correction for mobile robot homing. *Robotics and Autonomous Systems*, 50(1):41–49, 2005.

[30] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi. Divergent stereo in autonomous navigation: From bees to robots. *International Journal of Computer Vision*, 14(2):159–177, Mar. 1995.

[31] A. L. Shelton and J. D. E. Gabrieli. Neural correlates of encoding space from route and survey perspectives. *The Journal of Neuroscience*, 22(7):2711–2717, Apr. 2002.

[32] J. Shen and H. Hu. Visual navigation of a museum guide robot. In *Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA)*, volume 2, pages 9169– 9173, June 2006.

[33] R. Sim and G. Dudek. Learning environmental features for pose estimation. *Image and Vision Computing*, 19(11):733–739, 2001.

[34] L. Tang and S. Yuta. Vision based navigation for mobile robots in indoor environment by teaching and playing-back scheme. In *Proceedings of the International Conference on Robotics and Automation*, pages 3072–3077, May 2001.

[35] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings of the International Conference on Robotics and Automation*, pages 1023–1029, May 2002.

[36] S. Šegvić, A. Remazeilles, A. Diosi, and F. Chaumette. Large scale vision based navigation without an accurate global reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2007.

[37] J. Weng and S. Chen. Incremental learning for vision-based navigation. In *Proceedings of the IAPR International Conference on Pattern Recognition*, pages 45–49, 1996.

[38] J. Wolf, W. Burgard, and H. Burkhardt. Using an image retrieval system for vision-based mobile robot localization. In *Proceedings of the International Conference on Image and Video Retrieval (CIVR)*, pages 108–119, 2002.

[39] C. Zhou, Y. Wei, and T. Tan. Mobile robot self-localization based on global visual appearance features. In *Proceedings of the International Conference on Robotics and Automation*, pages 1271–1276, Sept. 2003.