

# Memory-Based Learning for Visual Odometry

Richard Roberts, Hai Nguyen, Niyant Krishnamurthi, Tucker Balch  
{richard, haidai, niyant, tucker}@cc.gatech.edu  
Center for Robotics and Intelligent Machines  
Georgia Institute of Technology, Atlanta, GA 30332, USA

**Abstract**— We present and examine a technique for estimating the ego-motion of a mobile robot using memory-based learning and a monocular camera. Unlike other approaches that rely heavily on camera calibration and geometry to compute trajectory, our method learns a mapping from sparse optical flow to platform velocity and turn rate. We also demonstrate an efficient method of computing high-quality sparse optical flow, and techniques for using this sparse optical flow as input to a supervised learning method. We employ a voting scheme of many learners that use subsets of the sparse optical flow to cope with variable dimensionality and reduce the dimensionality of each learner. Finally, we perform experiments in which we examine the learned mapping for visual odometry, investigate the effects of varying the reduced dimensionality of the sparse optical flow state, and quantify the accuracy of two variations of our learner scheme. Our results indicate that our learning scheme estimates monocular visual odometry mainly from points on the ground plane, and reflect to a degree the minimum dimensionality imposed by the problem. In addition, we show that while this memory-based learning method cannot yet estimate ego-motion as accurately as recent geometric methods, it is possible to learn, with no explicit model of camera calibration or scene structure, complicated mappings that take advantage of properties of the camera and the environment.

## I. INTRODUCTION

The problem of localization is often a fundamental challenge in mobile robotics, as tasks and algorithms therein usually make the assumption that the robot can be localized with respect to a global coordinate frame. Due to this assumption there is a repertoire of techniques and sensors whose goals are to facilitate reliable estimates of position. However, for each method there is usually a failure mode and operational requirement independent of the others. For example, differential GPS beacon systems, while extremely accurate, requires placement of extra beacons and are susceptible to physical obstructions as well as radio interference. These limitations can be mitigated by the addition of an inertial measurement unit which can provide accurate pose change measurements. However, IMU estimates of velocity and position normally drift significantly over time, while more accurate IMU units tend to be prohibitively expensive. Augmenting inexpensive IMUs with laser scan matching can be an effective method to boost accuracy, but scan matching can fail when there are no objects aligned with the laser and in range, or when the assumption of a static 2D environment does not hold. Laser range scanners also tend to be expensive, and their size and large power requirement prohibit their use on small outdoor robots. The limitations of each of the above techniques

normally necessitates fusing together estimates from several methods to create a reliable, robust estimate of pose that is less susceptible to any individual method’s failure mode.

Originally proposed by Matthies [1], visual odometry is rapidly becoming a popular pose estimation technique, as it can estimate the robot’s pose changes using only video containing distinctive features that are correlated with the robot’s motion. Visual odometry is useful for a variety of reasons, as cameras are fairly small and inexpensive, and can be mounted on even the smallest robots. Also, with rapid decreases in the cost and power requirement of computation, the expense of using visual odometry is often diminutive compared to other methods.

Even though there are existing visual odometry techniques based on geometric inference, which are highly accurate and work well in a variety of environments, we wish to explore visual odometry methods that are based on directly learning the mapping from optical flow to pose changes, without explicitly determining scene or camera geometry. As geometric visual odometry methods assume carefully calibrated parameters such as camera intrinsics and extrinsics as a first step, sensitive calibration procedures for determining pixel-perfect parameters often become a major challenge in the execution of these techniques. Furthermore, different camera and lens configurations require different sets of parameters, requiring algorithms to be modified accordingly. If these geometric assumptions can be relaxed or treated in a general way, then one visual odometry algorithm can be used without modifications across all cases.

We therefore propose a method for visual odometry using supervised learning, which makes significant progress towards overcoming these limitations. We start with the assumption that video with trackable features is available and that ground truth of the robot’s pose changes between frames can be obtained for training. Using this training set, we create a function approximator that learns a mapping from sparse optical flow to pose changes.

## II. RELATED WORKS

There has been a resurgence of interest in visual odometry algorithms as computers have become powerful enough to perform the steps needed for visual odometry at acceptable speeds. Work by Nister *et al.* [2] used 3D point tracking with preemptive RANSAC and the 5-point algorithm for essential matrix estimation. This technique has resulted in one of the most accurate and efficient visual odometry implementation

in a calibrated stereo pair setting, with accumulated errors of about 12 m over a 360 m course.

Ni and Dellaert [3] used an improved stereo-tracking method that determines feature displacement in both cameras then uses the matched features with the 3-point algorithm in a RANSAC harness. Agrawal and Konolige [4] scored RANSAC hypotheses in disparity space, then used Levenberg-Marquadt optimization to refine the hypothesis. Milella and Seigwart [5] used a robust iterative closest point technique to match points tracked 3D points. Dornhege and Kleiner [6] used an IMU to estimate the rotational motion component, and then had remaining vectors vote on whether the robot was stopped, moving forward, or moving backward.

Visual odometry has also enjoyed attention in the planetary exploration community, as GPS systems are only available on Earth. Helmick *et al.* [7] used a calibrated stereo pair to perform 3D point tracking with an initial motion hypothesis provided by wheel odometry. The 3D point correspondences between pair of frames were then fed into a coarse motion estimator, and refined using a probabilistic method. Corke *et al.* [8] used an omnidirectional camera and experimented with a robust sparse optical flow method that calculated motion changes by optimizing both the intrinsic parameters and motion, and compared this method to a structure from motion (SFM) algorithm.

Wang *et al.* [9] estimate monocular visual odometry through prior calibration of the location of the ground plane. Campbell *et al.* [10] also estimate accurate planar monocular visual odometry through prior manual calibration of the ground plane, and are able to detect precipices, or discontinuities, of the ground plane.

Another group, often less cited by computer vision researchers, interested in motion estimation using vision are biologists studying insect vision. Franceschini *et al.* [11], and Horridge and Longuet-Higgins [12] found that insects' compound eyes are arranged such that optical flow, above everything else, can be most efficiently estimated, as insects depend on perceiving motion to perform vital tasks such as obstacle and predator avoidance and landing. Methods for motion estimation suggested for insects, however, depend on highly parallelized neural networks, whose computational requirements exceed the capacity of current processors.

### III. METHODS

Our visual odometry method is comprised of a training stage that uses ground truth obtained separately, and a testing stage, in which ego-motion is computed from a video stream. During the training stage, our method records training instances that map the observed sparse optical flow vectors to the known forward velocity and turn rate of the platform, which we determine using laser scan matching. During the testing stage, the database of instances is queried using the sparse optical flow vectors observed at each frame, to obtain an estimate of the platform's velocity state. Because our platform is car-like, and not capable of moving to the side without turning, we only estimate forward velocity and

turn rate, but we expect that our method would generalize to three degrees of motion, as discussed later.

Briefly, our method works by first extracting sparse optical flow vectors, collecting them into a grid, and then training many learners that each work with a randomly-chosen subset of the grid cells. Voting among the learners during the testing stage handles the fact that individual grid cells sometimes do not contain any optical flow vectors.

A full explanation of our methods follows.

#### A. Sparse Optical Flow Computation

Our computation of sparse optical flow uses several modifications from baseline methods in order to improve efficiency and quality. However, our method of visual odometry does not assume any of these modifications, and also operates with simpler methods of optical flow computation.

The image is broken up into a grid, and at the first time step, the strongest Harris corner [13] in each grid cell is detected. For each subsequent time step, we use a KLT feature tracker [14] to track features into the next frame. While tracking, we fit a constant-image-velocity model to each feature's trajectory, and if a feature's motion over time deviates too much from this prediction, tracking of that feature is stopped. When tracking of a particular feature is stopped, the strongest Harris corner in the grid cell in which that feature was originally detected becomes a new feature, with a new motion model. This method provides sparse optical flow significantly faster than real time ( $>50$  Hz) on full-size video.

In addition, we use a temporal filtering step to prevent most unstable and mis-tracked features from becoming optical flow vectors. A feature being tracked is only reported as a sparse optical flow vector after it has been tracked for a number of frames.

For our trials we used a feature-detection grid cell size of approximately  $20 \times 20$  pixels, resulting in approximately 780 features being tracked in each image. We empirically determined this as the finest size that still allowed the entire process to run in real-time.

#### B. Mapping Sparse Optical Flow to Velocity State

We found that it was not feasible to learn a mapping directly from the entire set of sparse optical flow vectors to the output for two reasons. First, the dimensionality of the sparse optical flow state is too high for learning to generalize. Second, without a method for filling in missing portions of the optical flow field, the dimensionality would vary depending on the number and locations of the Harris corners detected in the image.

To address this problem, we use  $n$  (in our trials,  $n = 160$ ) separate k-Nearest-Neighbors (KNN) learners to estimate the current change in pose, with each learner taking as its input feature vector the average of the sparse optical flow vectors in each of  $m$  grid cells (chosen from a coarser grid than that of the feature tracking stage). In our trials, we used  $m = 4$  for experiments in which  $m$  was fixed. Each learner's grid cells are randomly selected when each learner is trained. Half

of the learners are trained to estimate forward velocity, while the other half estimate turning rate. Estimates from all of the learners are combined by averaging the  $l$  median values (in our trials,  $l = 3$ ) from the distribution of estimates, separately for forward velocity and turn rate. Learners missing data in one or more grid cells simply do not vote.

We selected  $m = 4$  empirically, using the results of Experiment IV-B. We also selected  $n$  and  $l$  empirically, although their effects on the accuracy of the final result were very small.

### C. Function approximation with KNN

Because we expect the mapping from optical flow to vehicle velocity state to be smooth, it is our desire that each KNN learner behave as a function approximator instead of a classifier. Therefore, for each prediction, our implementation fits a linear regression kernel to the  $k$  neighbors nearest to the observed input feature vector the hyper-plane that best maps the input feature vectors of those  $k$  neighbors to their output feature vectors, as described by Hastie *et al.* [15]. The output feature vector is then determined by evaluating the linear combination for the observed input feature vector.

We used  $k = 10$ , determined empirically, but varying  $k$  had little impact on visual odometry output. We implement KNN itself using a KD-tree [16].

### D. Tree migration

A variation of our method updates a score for each learner on-line, based on the deviation of its predictions from ground truth. This score, for each learner, is simply the mean-square difference between prediction and ground truth for forward velocity or turn rate (whichever value the learner is trained to predict), accumulated over the life of the learner.

Every so often (in our trials, every 2 seconds) a learner that is performing poorly relative to the other learners is deleted, and replaced with a random perturbation of a learner that is performing well. The learner to be replaced is chosen at random from a statistical distribution across the current set of learners, in which the weight of each learner in the probability density function (PDF) is proportional to the poorness of its score. Likewise, the learner whose perturbation is to serve as the replacement is selected from a distribution with the inverse PDF.

This process is a non-deterministic search for the subsets of the input dimensions that yield the most accurate function predictions, for a data set on which the learners were not trained. We refer to this process later as “tree migration”.

## IV. EXPERIMENTS AND DISCUSSION

We used four data sets in our experiments, comprised by three outdoor sets and one indoor set. The three outdoor sets were 4, 8, and 23 minutes in duration, and the indoor set was 2 minutes long. All data sets were collected using a steerable platform with two pivoting wheels at the rear, and two non-pivoting wheels at the front, carrying a SICK LMS-291 outdoor laser range finder, and a 2.33 GHz Apple

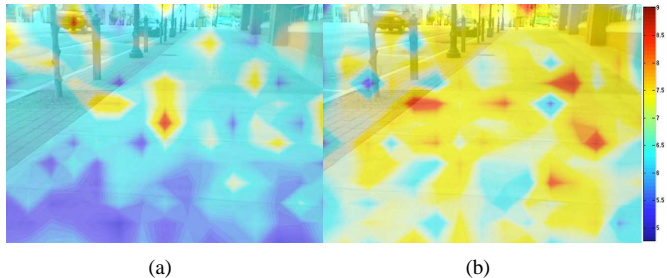


Fig. 1. For each image grid cell, the relative error for all of the learners making use of that grid cell, for a) forward velocity prediction errors, b) turn rate prediction errors. Colors range from blue (indicating small error) to red (indicating large error), as seen in the color scale on the right.

laptop with a built-in iSight camera. No wheel odometry was available.

The laser range finder was used to obtain approximate ground truth for training. Preliminary tests showed that trajectories obtained by laser scan matching were able to close loops of approximately one hundred meters with an error of only one meter, indicating that laser scan matching would be accurate enough to serve as approximate ground truth for training of, and comparison with, visual odometry. Images from the camera arrived at 30 Hz, at a resolution of  $640 \times 480$ . The camera was aimed downwards, such that only the upper 12% of the image was above the horizon. The platform was moved at speeds up to  $1.5m/s$ .

For every experiment, to ensure learning could generalize, visual odometry was trained on one of our data sets, and performance was evaluated while running on the others. The experiments we conducted examine the distribution of learner accuracies in the image, investigate the effect of varying the number of grid cells used by each learner, and evaluate the accuracy of our estimated visual odometry with and without tree migration.

The majority of our experiments involve comparing visual odometry trajectories to ground truth. Because visual odometry estimates and ground truth are both somewhat noisy, single-frame comparisons are not very meaningful. Therefore, to evaluate the accuracy of visual odometry, we break each sequence into short sub-sequences of fixed length (we used lengths of 2 and 4 s). Then, we average the forward velocities and turn rates from ground truth and visual odometry during each sub-sequence. Finally, errors computed for each sub-sequence are the differences between the average estimates of visual odometry and ground truth.

### A. Examining the Distribution of Learner Scores in the Image

To determine how accurately learners in various regions of the image estimate motion, an experiment was performed in which the output of each learner was compared to ground truth, with tree migration disabled. Then, each learner was examined in regards to the correlation between its average discrepancy from ground truth and the vertical and horizontal positions of its grid cells in the image.

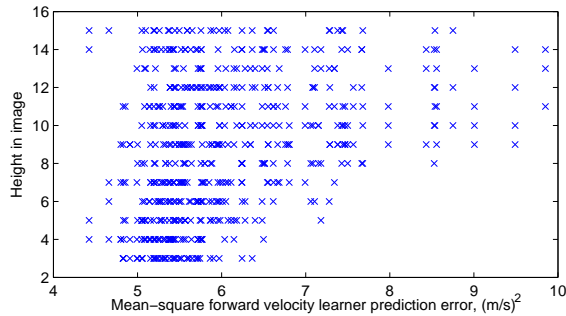


Fig. 2. Mean squared forward velocity prediction errors of each individual learner, plotted against height in the image.

For each grid cell, Figure 1 shows the average squared error for all of the learners making use of that grid cell, overlaid on an image typical of one of our outdoor data sets. These data, for every individual learner, are also plotted against height in the image for only velocity prediction in Figure 2.

These results indicate that the lower parts of the image yield more accurate predictions of both forward velocity and turning rate. We conjecture that the spatial locations, relative to the camera, of the points close to the camera and on the ground plane, are implicitly learned, and resolve the scale ambiguity inherent in monocular vision. Turning rate, on the other hand, is not subject to any scale ambiguity for distant points, as all stationary distant points move across the image at a rate dependent only on the rotation of the camera. Above and close to the horizon, however, large amounts of noise in optical flow due to moving objects cause large errors in all learners.

In monocular visual odometry, given a certain rate of divergence of the optical flow in an image, there is an ambiguity between the velocity of the camera, and the distance to the 3D points that comprise the scene structure. This ambiguity makes it impossible to tell the difference between a fast-moving camera observing far-away scene structure, and a slow-moving camera observing close-up scene structure. If, however, the orientation and position of the camera in relation to the ground plane is known, the distance to some scene structure lying on the ground plane is simply a function of its vertical position in the image. This information can resolve the ambiguity for planar motion.

While with geometric approaches to monocular visual odometry it is necessary to either make explicit assumptions about the structure of the scene or perform 3D reconstruction, our approach is able to implicitly take advantage of the information required to resolve those ambiguities. Similarly, we can generalize to suggest that learning mappings for other static scene structure, such as walls, would take place in the same way.

### B. Investigating the Effect of Varying Input Dimensionality

Results of visual odometry were compared against ground truth while the number of grid cells used by each learner was varied. These experiments were performed with tree

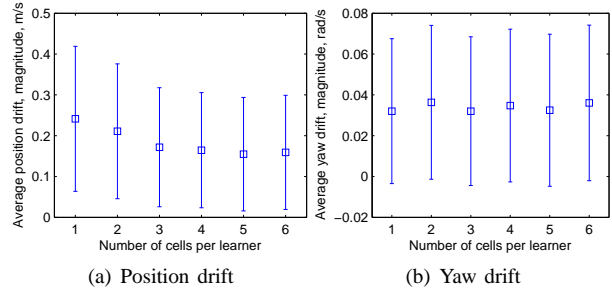


Fig. 3. Average drift from ground truth per unit time, (a) in position, and (b) in yaw, for learners utilizing various numbers of image grid cells. Error bars extend  $\pm 1$  standard deviation. Results are from running on the 23-minute outdoor data set, with learners having been trained on the 8-minute outdoor data set.

migration disabled. The 8-minute outdoor data set was used for training, and visual odometry was run on the 23-minute data set. The number of cells used by each learner was varied from 1 to 6.

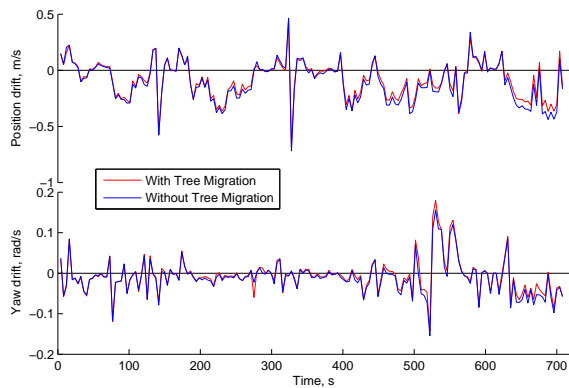
Figure 3 shows the drift, per unit time, of visual odometry from ground truth, for learners using numbers of grid cells ranging from 1 to 6. Errors were computed using the method described at the beginning of this section, using a sub-sequence length of 2 s. The absolute values of the forward velocity and turn rate errors were then averaged to obtain drifts, which reflect the approximate rate at which the trajectory computed using visual odometry deviates from ground truth.

Accuracy of visual odometry is poor when only 1 grid cell per learner is used, and improves until 3 grid cells are used. For larger number of grid cells, there is no further significant improvement in accuracy. There are only 2 degrees of freedom in our motion state, but the improved accuracy with three input dimensions may reflect either the necessity to disambiguate pitch changes from other motion components, an increased likelihood of most learners having at least two cells on the ground plane, or a robustness to noise. Further experiments would be required to determine the actual relationship between input and output degrees of freedom, but our results do demonstrate approximate correspondence therein.

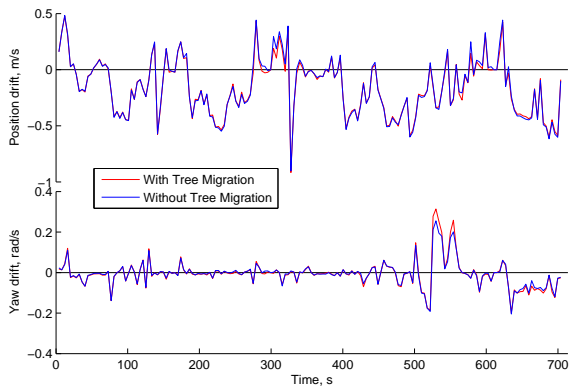
We expect our method to extend to almost any 3-DOF ego-motion estimation (such as that of a holonomic platform, a pan-tilt-roll camera, or even a pan-tilt-zoom camera), as long as scene structure remains fixed. Our method, in its current form, does not extend to 6-DOF motion, however, because this would require incorporating range measurements. Using stereo is interesting grounds for future work, however, as it raises the problem of representing the information gained in a way suitable for supervised learning.

### C. Evaluating Tree Migration

To evaluate whether tree migration improves the estimation of ego-motion, two experiments were performed. Visual odometry was trained with the indoor data set in one experiment, and with the 8-minute outdoor data set in the



(a) Run outdoors, trained outdoors

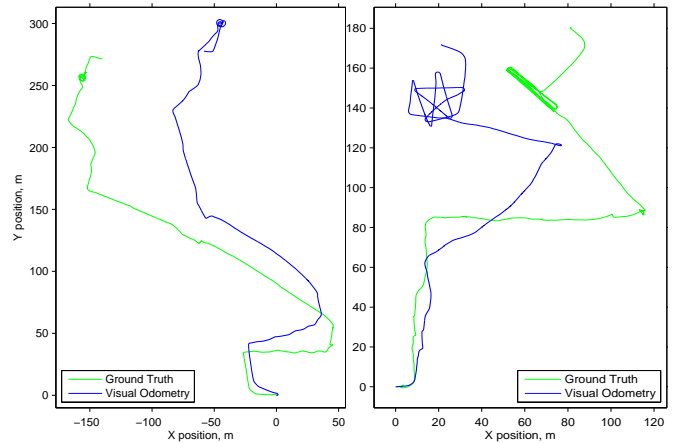


(b) Run outdoors, trained indoors

Fig. 4. Drift in the visual odometry trajectory over time on an outdoor data set, having been trained on (a) a different outdoor data set, and (b) the indoor data set. The blue line plots the drift with tree migration disabled, while the red line plots the drift with tree migration enabled. A small decrease in drift is observed, especially later in the run, with tree migration. Drift is estimated using the method described at the beginning of Section IV.

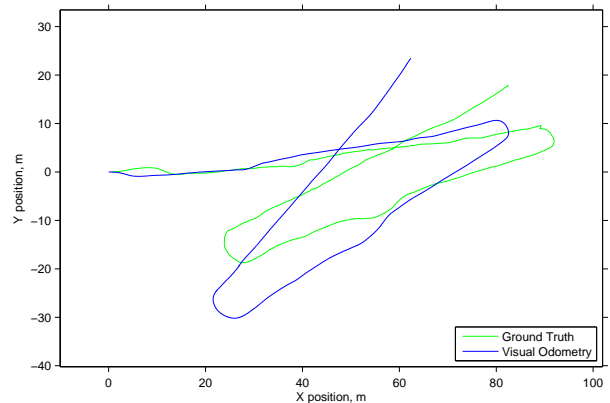
other. In both experiments, visual odometry was run on the 23-minute data set. We then evaluated the accuracy of visual odometry with and without tree migration as compared to ground truth, as described at the beginning of this section, using a sub-sequence length of 4 s. These results are shown in Figure 4.

While we hypothesized that tree migration would improve the results of visual odometry, very little difference in drift was actually observed between runs with and without tree migration. We believe that such a small effect was observed in part because of the way in which predictions from each learner are collected into a final prediction. After obtaining velocity and turn rate predictions from all learners, only the average of the 3 median predictions is used to build the trajectory, instead of the average of all predictions. We hypothesize that this method eliminates the effects of many of the outlying predictions, as long as they are a minority amongst all of the predictions. Under this assumption, predictions from learners using noisy or unreliable parts of the image are not likely to greatly affect the resulting trajectory, nor is removing them via tree migration likely to improve the results.



(a) 23 min. outdoor, first half

(b) 23 min. outdoor, second half



(c) 4-minute outdoor

Fig. 5. Platform trajectories: ground truth from laser scan matching, and estimated by visual odometry. Trajectories for (a) the first half, and (b) the second half, of the 23-minute outdoor run, and (c) the 4-minute outdoor run. In all cases, visual odometry was trained on the 8-minute outdoor data set.

#### D. Evaluating the Accuracy of Visual Odometry

Trajectories from visual odometry are shown and compared to the ground truth in Figure 5. As described previously, ground truth was obtained via laser scan matching. Tree migration was disabled when producing these trajectories, and 4 cells per learner were used. Trajectories depict visual odometry running on the 23-minute and 4-minute outdoor data sets, after having been trained on the 8-minute outdoor data set.

#### E. Performance

The amount of memory required for memory-based learning of ego-motion from sparse optical flow varied depending on the size of the training data set. A 2-minute data set at 30 Hz required approximately 100 MB of memory, while an 8-minute data set at the same frame rate required around 400 MB. Memory requirements scale linearly with the amount of training data. Given that the majority of training examples are likely duplicated, however, such as while driving straight at near-constant speed, these requirements could be

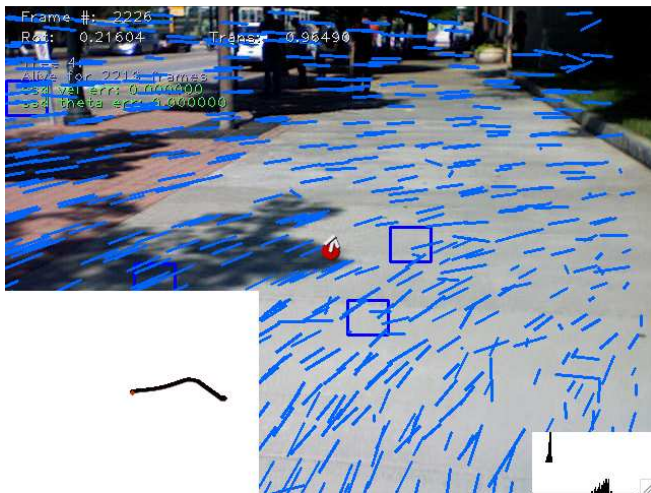


Fig. 6. A screen shot from our graphical display. The estimation of the current velocity and turn rate are shown in the top-left portion of the image, and indicated graphically by the arrow in the center. The accumulated trajectory is shown in the lower-left portion. The votes by each learner for velocity and turn rate are shown in the upper and lower histograms, respectively, in the lower-right. One individual velocity learner uses the grid cells outlined in blue, and its score is printed in the upper-left region of the image. Finally, the sparse optical flow is depicted throughout the image by light blue line segments.

greatly reduced by pruning similar examples during learner initialization. Computational performance was very good, as the entire visual odometry process, with  $640 \times 480$  images, including loading these images from disk, converting them from color to greyscale, performing feature tracking, and estimating motion, operated in real time ( $>30$  Hz).

A screen shot, in Figure 6, shows, for a single frame, the estimate of velocity and turn rate, the accumulated trajectory, the grid cells used by a single learner, the score of that learner, the histograms of learner votes for velocity and turn rate, and the sparse optical flow.

## V. CONCLUSION

We have developed a memory-based learning system that can give good estimates of the trajectory of a mobile robot. In doing so, we have demonstrated a method for efficient sparse optical flow computation, and a robust and elegant method for dealing with data of variable dimension. While our method is not yet as accurate as geometric methods, it demonstrates the viability of using a simple, purely learning

based method for visual odometry. Additionally, it illustrates that information about camera configurations and scene structure can be abstracted away by direct mapping from sensor inputs to the desired information.

## VI. ACKNOWLEDGMENTS

We would like to thank Alex Gray for consultation on machine learning methods. We would also like to thank our reviewers, whose comments were especially constructive and insightful.

## REFERENCES

- [1] L. Matthies, *Dynamic Stereo Vision*. PhD thesis, Carnegie Mellon University, 1989.
- [2] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 652–659, 2004.
- [3] K. Ni and F. Dellaert, "Stereo tracking and three-point/one-point algorithms – a robust approach in visual odometry," in *International Conference on Image Processing*, 2006.
- [4] M. Agrawal and K. Konolige, "Real-time localization in outdoor environments using stereo vision and inexpensive gps," in *ICPR*, 2006.
- [5] A. Milella and R. Siegwart, "Stereo-based ego-motion estimation using pixel tracking and iterative closest point," in *IEEE International Conference on Computer Vision Systems*, 2006.
- [6] C. Dornhege and A. Kleiner, "Visual odometry for tracked vehicles," in *IEEE International Workshop on Safety, Security and Rescue Robotics*, 2006.
- [7] D. M. Helmick, D. S. Y. C. Clouse, L. H. Matthies, and S. I. Roumeliotis, "Path following using visual odometry for a mars rover in high-slip environments," in *IEEE Aerospace Conference*, 2004.
- [8] P. I. Corke, D. Strelow, and S. Singh, "Omnidirectional visual odometry for a planetary rover," in *Proceedings of IROS*, 2004.
- [9] H. Wang, K. Yuan, W. Zou, and Q. Zhou, "Visual odometry based on locally planar ground assumption," in *Information Acquisition, 2005 IEEE International Conference on*, p. 6, 2005.
- [10] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa, "A robust visual odometry and preciptice detection system using consumer-grade single vision," in *ICRA*, 2005.
- [11] N. Franceschini, J. M. Pichon, and C. Blanes, "From insect vision to robot vision," in *Philosophical Transactions: Biological Sciences*, vol. 337, pp. 283–294, 1992.
- [12] G. A. Horridge and H. C. Longuet-Higgins, "What can engineers learn from insect vision?," in *Philosophical Transactions: Biological Sciences*, vol. 337, pp. 271–282, 1992.
- [13] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Fourth Alvey Vision Conference*, vol. 15, 1988.
- [14] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. Burkitt, "Performance of optical flow techniques," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1992.
- [15] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.
- [16] J. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," in *ACM Transactions on Mathematical Software*, vol. 3, pp. 209–226, September 1977.