Optical Flow Templates for Superpixel Labeling in Autonomous Robot Navigation

Richard Roberts and Frank Dellaert
Georgia Institute of Technology

***Abstract*— Instantaneous image motion in a camera on-board a mobile robot contains rich information about the structure of the environment. We present a new framework, *optical flow templates*, for capturing this information and an experimental proof-of-concept that labels superpixels using them. Optical flow templates encode the possible optical flow fields due to egomotion for a specific environment shape and robot attitude. We label optical flow in superpixels with the environment shape they image according to how consistent they are with each template. Specifically, in this paper we employ templates highly relevant to mobile robot navigation. Image regions consistent with *ground plane* and *distant structure* templates likely indicate free and traversable space, while image regions consistent with neither of these are likely to be nearby objects that are *obstacles*. We evaluate our method qualitatively and quantitatively in an urban driving scenario, labeling the ground plane, and obstacles such as passing cars, lamp posts, and parked cars. One key advantage of this framework is low computational complexity, and we demonstrate per-frame computation times of 20ms, excluding optical flow and superpixel calculation.**

I. Introduction

For a camera attached to a mobile robot, instantaneous image motion contains rich information about the robot's egomotion and the structure of the environment. In this paper, our contribution is to present a new framework for capturing this information, in which we attempt to address some shortcomings of previous work, and present an experimental proof-of-concept of this framework.

This framework, which we call *optical flow templates*, illustrated in Figure 1, permits semantically labeling image regions according to their observed optical flow and the predicted optical flow of several templates. Optical flow templates predict the optical flow due to robot egomotion, for a given robot velocity and attitude, and for a particular type of environment structure.

In this work, we use templates for *ground plane* and *distant structure*, and label regions that are not consistent with any other template as possible *obstacles*. Ground plane labels are likely to indicate fairly flat ground that can be driven upon. Far-away objects "at infinity" indicate line-of-sight directions that are likely free of obstacles. Image regions that are not consistent with either of these are likely to be nearby objects or other moving objects to be considered as obstacles.

The work to date towards using image motion for autonomous navigation has several drawbacks that limit its usefulness. While our framework does not completely solve the problem, it is a new way of looking at the problem that addresses some of the drawbacks.

In Section II we review related work and our position with respect to it, in Sections III and IV we explain optical flow templates and a method for labeling superpixels using them, and in Section V evaluate the method qualitatively, quantitatively, and in computational efficiency, in an urban driving scenario.

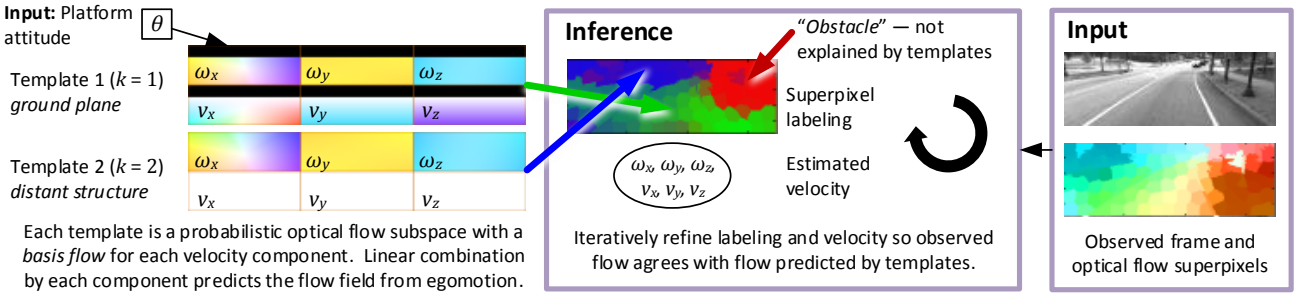


Fig. 1: *Optical flow templates* for superpixel labeling. Optical flow templates $W^k(\theta)$, one for each structure class *ground plane* ($k = 1$) and *distant structure* ($k = 2$), predict optical flow u_i at each i^{th} image location, given a platform velocity estimate $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^{\mathsf{T}}$ and attitude estimate $\theta \in \mathbb{SO}(3)$. Each template consists of 6 flow fields that combine linearly for each velocity component. For the optical flow color code, see Figure 3. Using observed optical flow, alternatively refine the labeling k_i for each i^{th} superpixel and the estimated velocity ξ. The special class $k = 0$ indicates optical flow pixels that cannot be explained by any template, which we label as *obstacle*.

II. Related Work

In autonomous driving, the current standard is to compute a 2D traversability map for path planning using information from 3D laser range scans, stereo correspondences, and structure from motion; for examples see [1], [2]. Becker *et al*. [3] accumulate optical flow information over short spans of time to infer a dense 3D reconstruction of the scene in front of the robot. The main drawback of these methods is the expensive computation required to calculate and then analyze large point clouds accumulated over many frames, both optimization problems with many variables. In comparison, our method uses very little computation because it directly estimates labels from optical flow data between pairs of frames, a single optimization problem with many fewer variables.

Instead of this standard method of measuring or computing 3D point clouds, other lines of research semantically label image regions using image appearance and machine learning [4], [5]. With applications outside of robotics as well, Hoiem *et al*. [6] also uses monocular features and machine learning to estimate 3D structure from single images. Our method, which uses optical flow information, is *complementary* to these methods that use image appearance. In this paper, our goal is to evaluate the utility of optical flow information alone, and to combine it with the appearance cues of this related work in future work.

Several lines of research combine 3D or image motion information with image appearance information using machine learning methods both to estimate semantic image labels and high-level semantic 3D structure. Brostow *et al*. [7] segment images into relevant regions such as street, sidewalk, car, *etc*. using structure-from-motion cues. Sturgess *et al*. [8] estimate similar segmentations using motion appearance and structure-from-motion information. These methods work well but require much labeled training data, and here we suggest that there is in fact a lot of information to be extracted from optical flow *before* having to use hand-labeled training data.

Geiger *et al*. [9] infer 3D street and traffic patterns from video from a moving platform, combining information from vehicle tracking, vanishing points, and image appearance. The information extracted is very rich, but comes at a high computational cost that makes it infeasible for small platforms.

Perhaps most closely related in method to this paper is that of Giachetti *et al*. [10], who use the differences between the observed optical flow and the flow predicted given motion on a ground plane to segment out other cars in the road. Additionally, Nourani-Vatani *et al*. [11] use optical flow for environment shape recognition with a discriminative learning method, matching flow fields to a database of locations using the flow field spatial statistics.

In contrast to planning over dense 2D or 3D maps, in which interpretation can be very difficult and computationally expensive, many researchers have investigated mobile robot control directly from optical flow. Inspired by research into the role of optical flow in animal and human navigation [12], [13], Duchon *et al*. [14] evaluate control laws for chasing, escaping, and other behaviors. Srinivasan *et al*. review a number of other bio-inspired optical flow control strategies, developed both by their group and others [15]. The main drawback of these methods is that they use heuristics, such as left-right flow balancing, that make their behavior difficult to predict and result in systematic errors. For that reason, we explicitly leverage a geometric model.

Conroy *et al*. [16] and Hyslop and Humbert [17] develop methods for autonomous robot control using "wide-field optic flow integration", which takes inner-products of the observed flow fields with a set of template flow fields. Using knowledge of possible coarse scene geometries such as walls and corridors, they develop templates and control laws. Our goal is different in that instead of inferring this coarse structure, we label finer structures and individual obstacles.

III. Optical Flow Templates

An optical flow template encodes the space of possible optical flow fields corresponding to a certain environment structure, invariant to the platform velocity. In turn, optical flow templates also specify a linear mapping from platform velocity $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^\mathsf{T}$ to optical flow, for a given robot attitude, and for a particular environment structure class. Shown in Figure 1, in this paper we specifically work with 3 possible classes: *ground plane*, *distant structure*, and *obstacle*. Let $W^k(\theta) \in \mathbb{R}^{2wh \times q}$ be the optical flow template for environment structure class k, where w and h are the image width and height, and q is the velocity dimension (in this paper $q = 6$). Optical flow templates are nonlinear functions of the platform attitude θ. Thus they predict a flow field for structure class k as $u^k = W^k(\theta)\,\xi$.

A. Optical Flow as a Gaussian Mixture of Templates

Because each pixel may be generated from any template, we model the optical flow u_i at each pixel i as a Gaussian mixture of the flow predicted $\hat{u}_i^k$ by each template,

$$p(u_i \,|\, \Lambda_i, \xi) = \mathcal{N}\left(0,\, \Sigma^0\right)^{\Lambda_i^0} \prod_{k=1}^{\kappa} \mathcal{N}\left(\hat{u}_i^k(\xi),\, \Sigma\right)^{\Lambda_i^k}, \quad (1)$$

where $\Lambda_i^k \in \{0, 1\}$ is a binary indicator of 1 if pixel i takes the *discrete* label k, and 0 otherwise. Σ is the covariance of the Gaussian noise on the optical flow consistent with the template, and Σ^0 is the covariance of zero-mean Gaussian noise on optical flow that is not consistent with any template, i.e. is labeled as *obstacle*. $\hat{u}_i^k(v) = W_i^k(\theta)\,\xi$ is the optical flow predicted by template k at pixel i, and we assume that the platform attitude θ is known, as explained in Section III-B where we calculate the templates. The notation W_i^k selects the pair of rows corresponding to pixel i from the optical flow template W^k. As is typical with notation in Gaussian mixtures, raising each term to the power of the indicator variable effectively makes only one term active for a given labeling.

Because the optical flow templates result in a linear relationship between velocity and optical flow, the measurement

likelihood in Eq. 1 is Gaussian, allowing inference to be fast and guaranteeing convergence. Each optical flow template is one instance of the optical flow subspace model introduced in our previous work [18].

B. Calculating Optical Flow Templates

The optical flow field in a calibrated projective camera is (see e.g. [19] for derivation)

$$u_i = \begin{bmatrix} y_i & \frac{x_i y_i}{f} & -f - \frac{x_i^2}{f} & \frac{x_i}{z_i} & \frac{-f}{z_i} & 0 \\ -x_i & f + \frac{y_i^2}{f} & \frac{-x_i y_i}{f} & \frac{y_i}{z_i} & 0 & \frac{-f}{z_i} \end{bmatrix} \xi, \quad (2)$$

where x_i and y_i are the horizontal and vertical image coordinates (with the origin at the image center) at pixel i, z_i is the depth at pixel i, and f is the focal length. The matrix is arranged such that $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^\mathsf{T}$ is the rotational and translational velocity in "robot coordinates", where the x-axis points forwards and the z-axis points down.

For a flow field u comprised of the concatenated flow vectors $u_i \in \mathbb{R}^2$, the optical flow template $W^k(\theta)$ is thus formed by stacking all of the matrices in Eq. 2. The depth z_i at each pixel depends on both the scene structure and the platform attitude θ. In this paper, we consider templates corresponding to *ground plane* and *distant structure*. For *ground plane*, we compute z_i using plane-line intersection geometry. For brevity, we omit the derivation which is a straightforward application of the tools in Chapter 2 of [20]. For *distant structure*, we use $z_i = \infty$ everywhere. In Figure 1, the ground plane template shows the camera is pitched slightly down – the boundary between the black and the colorful regions is the horizon.

In this paper, we assume the platform attitude θ is known. In our experiments, we obtain it from an attitude/heading reference system (AHRS). On autonomous ground vehicles and aircraft, this is common equipment, and is typically implemented using measurements from an accelerometer and gyroscope [21].

IV. Superpixel Labeling

Our method assigns a probability that each superpixel of optical flow in a frame of video belongs to each class, with each class represented by an *optical flow template.* Inference is a matter of labeling superpixels such that all superpixels assigned to the same template exhibit consistent optical flow within that template, and that all superpixels across all templates predict a consistent platform velocity. To simplify notation, we do not include frame numbers or time steps, but each frame is labeled independently.

Let $\lambda_j \in \mathbb{R}^\kappa$ be the vector of probabilities that superpixel j belongs to each class, and $\lambda_j^k \in \mathbb{R}$ be the k^{th} element, i.e. the probability that superpixel j belongs to class k. Let $k \in \{0, 1, 2\}$, where the special class $k = 0$ corresponds to *obstacle*, and $\kappa = 3$ is the number of classes (including *obstacle*). Both the superpixel labels λ_j and the velocity estimate ξ are alternatively refined, but for simplicity we omit iteration numbers from the notation.

A. Superpixel Labeling Given a Velocity Estimate

Our end goal is to label superpixels according to which optical flow template they are consistent with. This requires knowing the optical flow templates themselves (as obtained in Section III-B). Although each optical flow template encodes an optical flow subspace that is invariant to platform velocity, we want to enforce that all superpixels predict a consistent platform velocity. Thus, we will iteratively estimate the platform velocity (as explained in Section IV-B), and in this section, take the current velocity estimate ξ.

While in the previous Section III we defined the density of optical flow at a pixel i predicted by an optical flow template, we now switch to superpixels. All of the inference here may equally be performed at the pixel level, but using superpixels greatly reduces the computational expense. We use the index j to denote a superpixel, and use u_j and Λ_j to denote the flow and labeling of a superpixel, and $W_j^k(\theta)$ to denote the pair of rows corresponding to the pixel at the center of superpixel j.

We estimate the probability λ_j^k of each j^{th} superpixel belonging to each class. These probabilities define a multinomial distribution $p(\Lambda_j = e_k) = \lambda_j^k$ over the labels (where $e_k \in \mathbb{R}^\kappa$ is a vector with 1 in position k and 0 elsewhere, i.e. an assignment of the discrete indicator variables). Each assignment probability λ_j^k is the normalized likelihood that the superpixel is consistent with template k,

$$\lambda_j^k = \frac{l(\Lambda_j = e_k \,|\, \tilde{u}_j, \xi)}{\sum_{\bar{k}} l(\Lambda_j = e_{\bar{k}} \,|\, \tilde{u}_j, \xi)}, \quad (3)$$

where $l(\cdot) \propto p(\cdot)$ denotes a likelihood and $\tilde{u}_j$ is the *average* measured optical flow in superpixel j. We calculate the likelihoods here using Bayes' law,

$$l(\Lambda_j \,|\, \tilde{u}_j, \xi) = p(\tilde{u}_j \,|\, \Lambda_j, \xi)\, p(\Lambda_j), \quad (4)$$

where the measurement likelihood $p(\tilde{u}_j \,|\, \Lambda_j, \xi)$ is as in Eq. 1, except that the predicted flow $\hat{u}_i^k(\xi)$ is calculated by choosing pixel i to be at the center of superpixel j. $p(\Lambda_j)$ is the class prior, which in our experiments is a simple multinomial distribution.

In the next section, we describe refining the velocity estimate ξ given the labeling estimated with Eq. 3 above. As mentioned before, the velocity and labeling are alternatingly refined until convergence.

B. Refining the Velocity Estimate Given the Labeling

To refine the velocity ξ, we treat the labeling Λ_j as a hidden variable and update the velocity using expectation-maximization (EM),

$$\xi \leftarrow \arg\max_{\xi} \langle \mathcal{L}(\xi \,|\, \tilde{u}, \Lambda) \rangle, \quad (5)$$

where $\mathcal{L}(\cdot) = \log l(\cdot)$ denotes a log-likelihood, $\tilde{u}$ and Λ are the collections of optical flow vectors and indicator vectors for all superpixels, and the expectation $\langle \cdot \rangle$ is with respect to $p(\Lambda \,|\, \tilde{u}, \xi)$. Using Bayes' law and the linearity of the expectation, we have

$$\langle \mathcal{L}(\xi \,|\, \tilde{u}, \Lambda) \rangle = \mathcal{L}(\xi) + \sum_j \langle \mathcal{L}(\tilde{u}_j \,|\, \Lambda_j, \xi) \rangle, \quad (6)$$

where the expectation is now with respect to $p(\Lambda_j \,|\, \tilde{u}_j, \xi)$. To compute the expected log measurement likelihood $\langle \mathcal{L}(\tilde{u}_j \,|\, \Lambda_j, \xi) \rangle$, we note that the class probabilities λ_j^k calculated in the previous section comprise the *sufficient statistics* for this EM algorithm, giving everything we need to calculate

$$\langle \mathcal{L}(\tilde{u}_j \,|\, \Lambda_j, \xi) \rangle = \sum_k \lambda_j^k \mathcal{L}(\tilde{u}_j \,|\, \Lambda_j = e_k, \xi), \qquad (7)$$

where the the log measurement likelihood is the $\log$ of Eq. 1. Note that the velocity ξ used in Eq. 1 is in fact the variable being optimized for in this section, and is not fixed here as it was in Section IV-A.

Because the measurement likelihood is Gaussian, the expected log-likelihood is a quadratic, and thus refining the velocity estimate with the maximization in Eq. 5 is a linear least-squares problem that is easily solved using direct methods.

C. Summary and Implementation of Method

Given each of the components we have introduced in this section, we now summarize the steps that take place for each incoming video frame here in Algorithm 1:

Algorithm 1: Superpixel labeling for each frame.

Input: Current and previous video frames.
Input: Platform attitude θ (e.g. from AHRS).
Result: Class label probabilities λ_j^k for each superpixel j and class k.

1 For current frame, compute SLIC superpixels [22].
2 For previous and current frames, compute TV-L^1 [23], [24] optical flow $\tilde{u}$ and average flow $\tilde{u}_j$ in each superpixel.
3 Compute basis flows $W_j^k(\theta)$, at image locations corresponding to superpixel centers.
4 Initialize class label probabilities uniformly, $\lambda_j^k = \frac{1}{\kappa}$.
5 **while** *change in* $\langle \mathcal{L}(\xi \,|\, \tilde{u}, \Lambda) \rangle$ *is greater than convergence threshold*[1] ϵ **do**
  Update $\xi \leftarrow \arg\max_\xi \langle \mathcal{L}(\xi \,|\, \tilde{u}, \Lambda) \rangle$ (Sec. IV-B)
  Update $\lambda_j^k \leftarrow \frac{l(\Lambda_j = e_k \,|\, \tilde{u}_j, \xi)}{\sum_{\bar{k}} l(\Lambda_j = e_{\bar{k}} \,|\, \tilde{u}_j, \xi)}$ (Sec. IV-A)
end
6 **return** λ_j^k

V. Experiments

Our experimental platform is a car equipped with a Point Grey Flea3 (gigabit ethernet version) camera running at 1380×480 at 10Hz, and a MicroStrain 3DM-GX3-45 inertial navigation system (INS), from which we use only the AHRS attitude estimate. Both were connected to an Intel Core i7 laptop for data logging, and the computations described in this paper were performed on the logged data.

[1]This convergence criteria relies on the likelihood normalization constant remaining constant during optimization. To accomplish this, it is sufficient to include in the likelihood calcaulations the normalization constants from all Gaussians involving the velocity v.

We operated the car in an urban environment. In collecting the dataset, we took care not to follow any car in front too closely. We did not make any other special considerations in driving style while collecting the dataset.

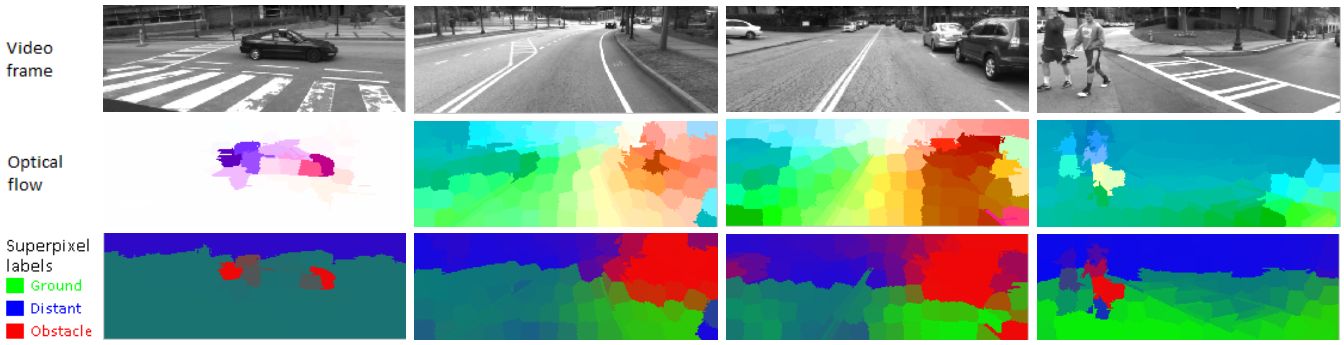
The parameters $\Sigma = \mathrm{diag}(7.0)$ and $\Sigma^0 = \mathrm{diag}(15.0)$ were used for the noise covariances. These values were selected empirically: Too tight covariance causes too many superpixels to be labeled as *obstacle* due to slight noise in optical flow measurements, while too loose covariance causes all superpixels to be labeled only *ground plane* or *distant structure*. For the label priors, we used $p(\Lambda = \textit{ground plane}) = 0.4$ and $p(\Lambda = \textit{distant structure}) = 0.4$ for superpixels below the horizon, and $p(\Lambda = \textit{ground plane}) = 0$ and $p(\Lambda = \textit{distant structure}) = 2/3$ above the horizon (in each case the remaining probability is for *obstacle*). These values were chosen by manual estimation of the image portion occupied by each class in several typical frames, although the algorithm is not very sensitive to this prior.

Our implementation of the algorithms is in C++ using the GTSAM factor graph library [25]. The C++ classes are wrapped in MATLAB using the `wrap` utility included in GTSAM, and we perform data loading, scripting, and visualization in MATLAB. All source code and datasets will be made available online on the authors' web site by the time of publication (link will be provided in camera-ready version).

A. Qualitative Analysis

In Figure 2, we present examples of the labeling produced by our method that we consider successful. These examples are successful because the information they contain is useful for autonomous navigation. For the most part the major structure above the ground plane and close to the camera is labeled as "*obstacle*" (red), and the *ground plane* (green) and *distant structure* (blue) are mostly correctly labeled. The *obstacle* structure above the ground plane is often obstacles, road boundaries, or independently-moving objects. Such structure could be avoided by a navigation method or passed on to higher-detail vision processing.

Sometimes the labeling produced by our method contains errors. One type of error arises because the optical flow estimate is incorrect. Most frequently, this occurs in regions with smooth or repetitive texture, as shown in Figure 4a. This type of error points us towards future work because the root cause of this problem is computing optical flow as an *input* that is unaware of the global optical flow constraints provided by our optical flow templates. We expand on this in the discussion. Another type of error occurs when the differences between the predicted optical flow from two templates becomes too small to distinguish from noise, as shown in Figure 4b. In this case, superpixels on the ground plane may be be labeled with 50% probability of belonging to either the *ground plane* or *distant structure* classes, indicated in the figure by the green/blue or turquoise blended color; also, ground plane superpixels may be labeled as *distant structure* if small error in the velocity estimate of our method causes



(a) Car crossing perpendicularly while platform stationary (b) Passing lamps on right side while driving forwards (c) Passing cars parked on both sides while driving forwards (d) Encountering pedestrians during a right turn

Fig. 2: Example video frames, superpixel optical flow fields, and superpixel labels by our method.

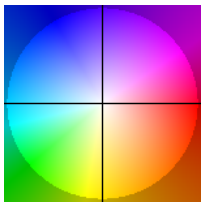
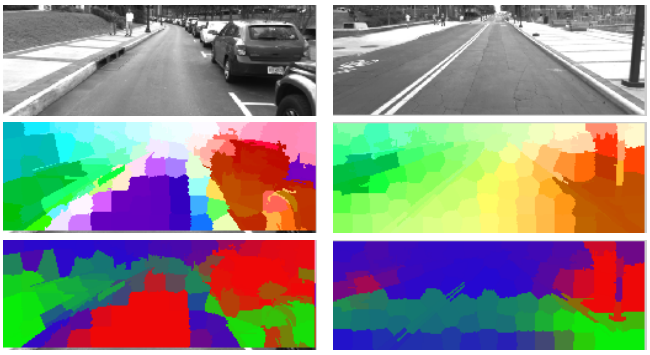


Fig. 3: The optical flow color code, as in [26]. Flow fields (e.g. Figure 1) are displayed with a color at each pixel. The hue indicates the direction of the flow and the saturation its magnitude. Here, the center of the black cross (color white) is zero flow. Yellow is downwards flow, red rightwards, etc.



(a) Error in labeling the ground as *obstacle* due to optical flow errors caused by smooth texture. (b) Error in labeling ground plane as *distant structure* due to similarity between rotational and translational basis flows.

Fig. 4: Examples demonstrating errors in labeling by our method.

the optical flow on the ground plane to agree more closely with rotational flow than with translational flow.

B. Quantitative Analysis

We hand-evaluated 200 frames of video and labels produced by our method, the results of which are shown in Figure 5. In each frame, we determined the number of objects or regions of environment structure mislabeled by our method. "Extra obstacles" are image regions that our method labels in the *obstacle* class, yet there is no structure above

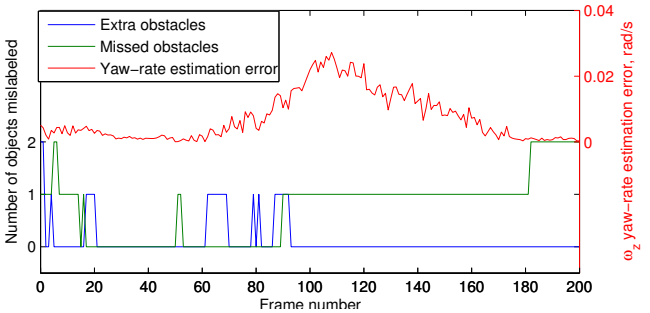


Fig. 5: Number of objects mislabeled by our method, evaluated by hand-inspecting 200 frames labeled by our method. We counted "extra obstacles", which are image regions detected by our method as the *obstacle* class when no nearby structure was in fact present, and "missed obstacles", where nearby structure was present but not detected.

the ground plane near the camera in that region. "Missed obstacles" are image regions containing structure above the ground plane near the camera but that our method does not label as *obstacle*. For example, in Figure 4a we would count the region mislabeled on the ground plane as one "extra obstacle", and a missed car or pedestrian would count as one "missed obstacle". In hand-labeling, we took into account the smoothing inherent in the optical flow calculation, so several objects close together, for example poles on the side of the road, are only counted as one object.

Because our method estimates superpixel labels jointly with platform velocity, we test whether errors in superpixel labels are coupled with errors in yaw-rate estimation. Figure 5 plots the yaw-rate error of our method as compared with the vehicle's gyroscope. While gross yaw errors would certainly be coupled with many incorrect superpixel labels, the comparison demonstrates that there is no correlation between small yaw errors and errors in superpixel labeling.

C. Timing

Our research implementation is single-threaded, and takes approximately 20 ms per frame (± 2 ms) to infer superpixel labels and velocity, already having the optical flow and

superpixel segmentation available as input. For each frame, the combined time to calculate TV-L^1 optical flow and SLIC superpixels on the CPU is $500\,\mathrm{ms}$ per frame, however, there are GPU versions available of both algorithms that achieve real-time performance using CUDA. All timings were measured on an Intel Core i7 3.4 GHz desktop. The nature of the linear least-squares problem makes it adaptable to parallelization and GPU implementation.

VI. Summary and Future Work

We have presented a new framework for interpreting optical flow observed by a mobile robot, called *optical flow templates*. Using templates for *ground plane* and *distant structure*, our method labels image regions whose flow is consistent with these templates, as well as labeling flow inconsistent with either as *obstacle*. This latter class comprises objects that occupy space above the ground plane near to the robot, and may be passed on for more detailed and computationally intensive processing.

The key aspects of the superpixel labeling method using this framework, in relation to previous work, include that computational complexity is very low, and geometric models of optical flow remove the need for hand-labeled training data or heuristics. We present an experimental proof-of-concept, labeling video in an urban driving scenario.

One direction of future work is to jointly infer optical flow along with the velocity and superpixel labels. We have observed, as shown in Section V-A, that many of the errors made by our method come from errors in optical flow estimation. These errors especially occur in regions of smooth texture, where the task of optical flow is underconstrained and ill-posed without a global optical flow model. Our optical flow templates in fact provide such a model, so joint inference should make much more accurate labels possible. Another benefit of joint inference will be sharper image segmentations of objects and boundaries between classes, which are currently blurred due to the smoothness terms necessary to compute dense optical flow.

References

[1] J.-F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *Journal of Field Robotics*, vol. 23, no. 10, pp. 839–861, Oct. 2006.

[2] A. Huertas, L. Matthies, and A. Rankin, "Stereo-Based Tree Traversability Analysis for Autonomous Off-Road Navigation," *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, pp. 210–217, Jan. 2005.

[3] F. Becker and F. Lenzen, "Variational recursive joint estimation of dense scene structure and camera motion from monocular high speed traffic sequences," *International Conference on Computer Vision*, 2011.

[4] J. Michels, A. Saxena, and A. Y. Ng, "High speed obstacle avoidance using monocular vision and reinforcement learning," in *Proceedings of the 22nd international conference on Machine learning - ICML '05*. New York, New York, USA: ACM Press, 2005, pp. 593–600.

[5] Y. N. Khan, P. Komma, and A. Zell, "High resolution visual terrain classification for outdoor robots," pp. 1014–1021, 2011.

[6] D. Hoiem, A. A. Efros, and M. Hebert, "Recovering Surface Layout from an Image," *International Journal of Computer Vision*, vol. 75, no. 1, pp. 151–172, Feb. 2007.

[7] G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," *European Conference on Computer Vision*, 2008.

[8] P. Sturgess, K. Alahari, L. Ladicky, and P. H. S. Torr, "Combining Appearance and Structure from Motion Features for Road Scene Understanding," *Procedings of the British Machine Vision Conference 2009*, pp. 62.1–62.11, 2009.

[9] A. Geiger, M. Lauer, and R. Urtasun, "A generative model for 3D urban scene understanding from movable platforms," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Ieee, Jun. 2011, pp. 1945–1952.

[10] A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for road navigation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 34–48, 1998.

[11] N. Nourani-Vatani, P. V. K. Borges, J. M. Roberts, and M. V. Srinivasan, "Topological localization using optical flow descriptors," *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1030–1037, Nov. 2011.

[12] J. J. Gibson, "Visually controlled locomotion and visual orientation in animals." *British journal of psychology*, vol. 49, pp. 182–194, Apr. 1958.

[13] M. Lehrer, M. V. Srinivasan, S. W. Zhang, and G. A. Horridge, "Motion cues provide the bee's visual world with a third dimension," *Nature*, vol. 332, no. 6162, pp. 356–357, Mar. 1988.

[14] A. Duchon, W. H. Warren, and L. P. Kaelbling, "Ecological Robotics: Controlling Behavior with Optic Flow," in *International Joint Conference on Artificial Intelligence*, 1995.

[15] M. Srinivasan, S. Thurrowgood, and D. Soccol, "Competent Vision and Navigation Systems," *IEEE Robotics & Automation Magazine*, vol. 16, no. 3, pp. 59–71, Sep. 2009.

[16] J. Conroy, G. Gremillion, B. Ranganathan, and J. S. Humbert, "Implementation of wide-field integration of optic flow for autonomous quadrotor navigation," *Autonomous Robots*, vol. 27, no. 3, pp. 189–198, Aug. 2009.

[17] A. M. Hyslop and J. S. Humbert, "Autonomous Navigation in Three-Dimensional Urban Environments Using Wide-Field Integration of Optic Flow," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 1, pp. 147–159, Jan. 2010.

[18] R. Roberts, C. Potthast, and F. Dellaert, "Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[19] D. Heeger and A. Jepson, "Visual Perception of Three-Dimensional Motion," *Neural Computation*, vol. 2, pp. 127–137, 1990.

[20] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[21] J. Farrell, *Aided Navigation: GPS with High Rate Sensors*. McGraw-Hill, 2008.

[22] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–82, Nov. 2012.

[23] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L 1 optical flow," *Ann. Symp. German Association Patt. Recogn*, 2007.

[24] J. Sánchez, E. Meinhardt-Llopis, and G. Facciolo, "TV-L1 Optical Flow Estimation," *Image Processing On Line*, 2012.

[25] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Tech. Rep. GT-RIM-CP&R-2012-002, 2012.

[26] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A Database and Evaluation Methodology for Optical Flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, Nov. 2010.