Reducing Adaptation Latency for Multi-Concept Visual Perception in Outdoor Environments

Maggie Wigness*, John G. Rogers III*, Luis Ernesto Navarro-Serment†, Arne Suppe† and Bruce A. Draper‡

*U.S. Army Research Laboratory †Robotics Institute, Carnegie Mellon University ‡Colorado State University

Abstract— **Multi-concept visual classification is emerging as a common environment perception technique, with applications in autonomous mobile robot navigation. Supervised visual classifiers are typically trained with large sets of images, hand annotated by humans with region boundary outlines followed by label assignment. This annotation is time consuming, and unfortunately, a change in environment requires new or additional labeling to adapt visual perception. The time is takes for a human to label new data is what we call *adaptation latency*. High adaptation latency is not simply undesirable but may be infeasible for scenarios with limited labeling time and resources. In this paper, we introduce a labeling framework to the environment perception domain that significantly reduces adaptation latency using unsupervised learning in exchange for a small amount of label noise. Using two real-world datasets we demonstrate the speed of our labeling framework, and its ability to collect environment labels that train high performing multi-concept classifiers. Finally, we demonstrate the relevance of this label collection process for visual perception as it applies to navigation in outdoor environments.**

I. INTRODUCTION

Accurate environment perception is critical for autonomous robots to plan paths on traversable terrain and avoid object collision during navigation. While many sensors have been used to help with perception [1], [2], [3], [4], speedups in image processing have allowed vision-based perception to emerge in mobile robots [5], [6], [7], [8], [9], and benefits path planning because visual data allows robots to perceive a large area of the environment at once.

In general, high performing supervised visual classifiers require large sets of training data to incorporate variations in illumination, perspective, occlusion and appearance. For example, state of the art object classifiers are trained with over one million images [10], [11]. Although raw visual data is easy to collect, labeling this data can be time consuming as it requires human intervention to assign semantic labels to training instances. This process is even more demanding for scene labeling classifiers [12], [13] because distinct regions in images must be outlined before assigning labels.

To ensure the highest quality visual perception, training data should be collected from the environment where navigation tasks will be performed. Thus, each domain change requires new data collection and labeling. We define the time for a human to label a new set of training data as *adaptation latency*. This represents the time robots are unable to navigate autonomously because perception models are being adapted.

Adaptation latency has yet to be discussed in existing supervised multi-concept visual perception systems used in robotics applications [1], [5], [6], [7]. Annotation of images is performed as a necessary, but time consuming step to train supervised classifiers. However, in scenarios with limited time and resources, supervised annotation of large image sets may be infeasible. Unsupervised or self-supervised approaches have been used to eliminate labeling effort [3], [9], [14], [15], [16], [17], but produce a limited environment vocabulary, e.g., *traversable* versus *non-traversable*. These techniques do not generalize well to more complex navigation tasks that require a richer set of scene semantics, such as verbal navigation commands from humans [18].

Our work is motivated by scenarios that need more than a binary understanding of environments, and that have limited time and resources to collect this information. In this paper, we discuss an efficient labeling framework called Hierarchical Cluster Guided Labeling (HCGL) [19] that reduces adaptation latency without significantly compromising visual perception. HCGL uses unsupervised learning to segment and cluster visual data to quickly label groups of data. Group labeling reduces adaptation latency in exchange for a slight degradation in label accuracy. Although label noise may impact classifier learning, we show that visual perception trained with HCGL allows for reliable path planning and successful navigation. HCGL is compared to a fully supervised labeling approach by evaluating pixel labeling rate, pixel-wise classification and autonomous navigation via road terrain with respect to adaptation latency.

In summary, this paper makes several contributions. First, we introduce an efficient real-world feasible image labeling framework to the robotics and environment perception domains. Second, we show that trading greater efficiency for minimal training label noise does not significantly degrade visual perception learning. Finally, we present the first navigation task-based evaluation of multi-concept visual perception with respect to adaptation latency.

II. REDUCING ADAPTATION LATENCY

Supervised label collection produces high quality labeled data, but is time consuming for two reasons: 1) training sets are typically large, and 2) images capture multiple terrains and objects in the scene that need to be localized before label assignment. Prior to this work, image annotation tools such as LabelMe [20] have been used to facilitate supervised

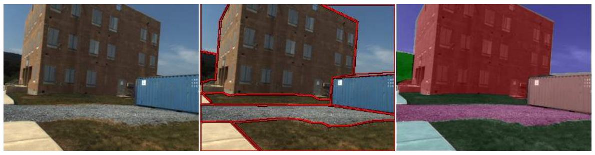


Fig. 1: Example of supervised labeling input (left), require outlining of regions (center) and the final label output (right).

labeling. LabelMe allows annotators to precisely outline, via mouse clicks, and assign labels to each distinct region. Fig. 1 is an example of a training image (left), required outlining (middle) and labeled output (right - see class/color legend in Fig. 9) using LabelMe. Labeling 250 images requires over 20 hours of effort (discussed in Section III), causing high latency during domain changes and inhibits fast adaptation.

The goal of this work is to train supervised multi-concept visual classifiers using large amounts of labeled environment data with limited human interaction. We use the Hierarchical Cluster Guided Labeling (HCGL) framework, and introduce several modifications to better suit real-world environment data. An overview of HCGL is provided, but we refer the reader to [19] for further details and motivation of the framework. After discussing our efficient label collection technique, we compare HCGL to supervised labeling with LabelMe to demonstrate the speedup achieved.

A. HCGL Overview

HCGL leverages unsupervised learning to reduce labeling effort. Specifically, HCGL hierarchically clusters data into groups, and annotators label multiple training instances at once by assigning a single label to each selected group. The middle of Fig. 2 is an illustration of a hierarchy created by HCGL. Each node is a group of data, and colors depict which class most instances represent. Black wedges indicate the percentage of noise in each group, i.e., images not representing the dominating class.

Hierarchical clustering generalizes to domains without any a priori knowledge since the number of groups is not specified in advance. It also eliminates additional latency introduced by other group labeling approaches that iteratively re-cluster data [21], [22]. However, the hierarchy is large and encodes coarse and fine-grained feature similarities. For example, clustering data from environment A (details in Table I) produces groups higher in the hierarchy that contain *gravel* and *asphalt* examples because they share coarse-grained similarities that map to a more general concept like *ground*. Finer-grained differences allow these classes to group independently lower in the hierarchy. HCGL defines an interestingness measure to locate the transition between coarse and fine-grained groupings, which establishes a subset of groups from the hierarchy that can be quickly labeled.

The interestingness measure compares structural change between a cluster, c, and its parent, p. Arrows connecting nodes in Fig. 2 denote the c and p relationship. Internal structure of a group is modeled through the eigendecomposition of the covariance matrix of its images, and structural change is represented as the angle between the primary directions of variance, v_c and v_p, of c and p respectively. Formally, the interestingness of c is defined as the cosine distance:

$$\Delta(c) = 1.0 - \langle v_c, v_p \rangle \qquad (1)$$

Groups with greatest interestingness are selected for labeling.

B. Multi-Concept Environment Data

HCGL was designed to cluster and label single-concept images. To generalize to multi-concept environment data, without additional human effort, images are first automatically segmented to create disjoint regions that are treated as individual training instances. While many segmentation techniques could be used to generate input for HCGL, we use SLIC [23] to over-segment each training image into approximately 150 segments, which was selected empirically (seen in the left of Fig. 2). Segments are clustered in HCGL using LAB color histograms, LBP texture features, a 200 word SIFT codebook and normalized region coordinates. Unlike supervised approaches that label a single image at a time, HCGL labels fragments of multiple training images simultaneously (seen in the right of Fig. 2).

C. Selections and Labeling

Interestingness scores help localize areas of the hierarchy that should be labeled, but other ordering heuristics can help emphasize HCGL labeling objectives. These objectives include collecting labels quickly, discovering the underlying concepts and assigning accurate labels. The original implementation of HCGL laid out three ordering heuristics to emphasize these objectives. They are summarized as:

1) Interestingness - the degree of structural change seen between related nodes in the hierarchy
2) Exploitation - the number of samples assigned labels during a single query
3) Exploration - the likelihood of discovering a concept different from those previously labeled

Briefly, interestingness is defined in (1), exploitation is the number of training samples in a group (higher scores for groups higher in the hierarchy), and exploration is defined by the path length between groups (higher scores for groups further from one another in the hierarchy).

Experiments performed on single-concept image benchmark datasets [19] showed minimal classification performance differences when comparing the ordering criteria independently. However, those datasets had a uniform distribution of classes. Real-world environments typically exhibit a non-uniform distribution of classes across pixels. Thus, classes with more pixels may be favored by some criteria.

To balance the objectives, we linearly combine these criteria for our experiments to produce a multi-objective ordering score. Every group is ranked according to each criterion and a weighted sum of these rankings make up the final score for a group. The unlabeled group with the largest rank score is selected as the next labeling query. For all experiments, the three criteria are weighted equally.

As previously mentioned, HCGL trades some label accuracy for greater labeling efficiency. Label noise is introduced

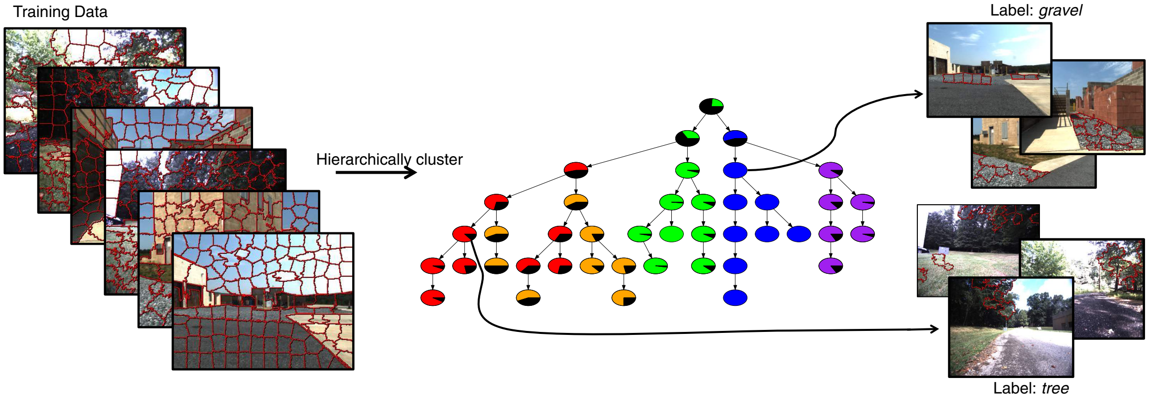


Fig. 2: Visualization of HCGL on multi-concept environment data. Training data is over-segmented (left), segments are hierarchically clustered (center), and clusters from the hierarchy are selected, displayed to the user and assigned the majority concept label (right).

TABLE I: Details of environment datasets.

Environment	# Training Images	Label Set
A	274	*asphalt,building,concrete* *grass,gravel,object,sky,tree*
B	1,982	*building,grass,object,road* *sidewalk,sky,tree*

when a selected group contains images from multiple classes and the majority label is assigned to the data. Fig. 2 illustrates majority labeling with the group labeled *tree* since it also contains regions that are actually *grass*. The idea is that minimal label noise will not significantly inhibit classifier learning when combined with a large set of accurately labeled data. In some cases, there may not be a clearly defined majority concept and the user may label the group *mixed* which produces no label information for that query.

III. Speed and Pixel-Wise Classification Evaluation

We use two real-world environments to demonstrate the speed and performance of HCGL when collecting labels for multi-concept visual perception. The environments are outdoor urban training facilities with multiple terrain types, buildings, cars and other objects. Training data for environment A was collected using a high dynamic range camera at a previous experiment performed in 2012. Images were taken at 5 different time blocks over two days from 53 locations in the environment [24]. Training data for environment B is captured via teleoperation using the robot described in Section IV-B. Environment B contains significantly more training images than environment A because it is the combination of three training sets collected on consecutive days under varying weather conditions. Performance on this dataset shows how HCGL is able to scale with increasing training set sizes. An overview of the datasets is provided in Table I and example images are seen throughout the paper.

We compare HCGL to the supervised labeling baseline LabelMe, where training images are labeled in random order. Pixel-wise labeling and classification accuracy are evaluated as a function of labeling interaction time (i.e., adaptation latency) to show the speed at which techniques can collect multi-concept scene labels for visual classifiers.

A. Labeling Speed and Label Accuracy

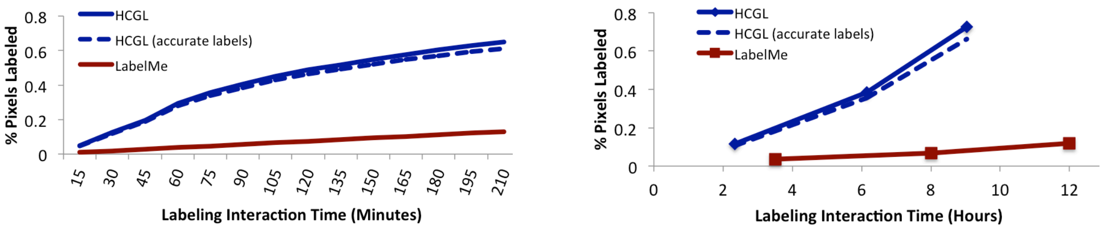
The first evaluation compares the speed at which HCGL and LabelMe assign labels to the training set. Fig. 3 shows the percentage of labeled pixels as a function of labeling interaction time. For both datasets, HCGL collects six to seven times the amount of label information as LabelMe at any given point in the labeling process. Thus, HCGL provides supervised classifiers with significantly more data for learning after limited labeling time. Interaction time for environment B is on the order of hours because the three training sets were labeled separately and then combined.

Collecting labels quickly is an important objective, but recall that HCGL achieves this speed by trading some label accuracy with majority labeling. The dashed blue lines in Fig. 3 show the percentage of pixels that received accurate labels from HCGL (determined using labels collected with LabelMe). This line represents $\sim 5-10\%$ pixel label noise; a small fraction for a large gain in efficiency.

B. Pixel-Wise Classification

Next, labels collected from HCGL and LabelMe are compared by training visual classifiers and evaluating pixel-wise classification accuracy on a disjoint test set. We use the Hierarchical Inference Machine (HIM) [13], an approach for scene parsing and region classification. HIM decomposes images into a hierarchy of nested superpixels and incorporates both feature descriptors and contextual cues. HIM trains a hierarchy of regressors that predict the label distribution for pixels in each superpixel region at a coarse level of segmentation, and use that information to refine predictions at a finer level of segmentation with greater spatial locality.

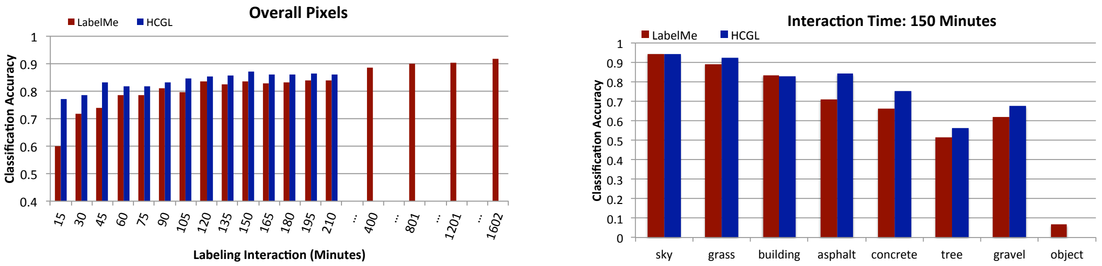
In our experiments, the predictor is a decision forest regressor with 10 trees, and the F-H [25] algorithm is used to create a 7-level segmentation hierarchy. Features include SIFT codebooks, LAB colorspace statistics, texture information and statistics on superpixel size and shape. The HIM was selected because of its on-line processing and ability to



(a) Environment A

(b) Environment B

Fig. 3: Labeling rate for HCGL and LabelMe for two training sets, and accuracy of HCGL label assignment (dashed lines).



(a) Overall pixel classification accuracy

(b) Per-class classification accuracy

Fig. 4: Pixel-wise classification accuracy on the environment A test set. (a) Overall pixel classification as a function of interaction time. (b) Classification accuracy by class after 150 minutes of labeling interaction time. Classes are ordered by their pixel-wise frequency in the data.

interface with our robot platform (discussed in Section IV-B). HIM processes a 640 × 384 image in approximately 2 seconds on a dedicated quad-core i7-3615QM at 2.3 GHz, with feature extraction being the dominant cost. A rigorous performance assessment of this algorithm was conducted during an earlier field trial [24].

Fig. 4a shows the overall pixel accuracy for environment A (the only dataset with a large, labeled test set [24]). Although HCGL introduces a small amount of label noise, the larger volume of labeled data allows HCGL to train higher performing classifiers than LabelMe through 210 minutes of labeling interaction. HCGL labeling is terminated at this point to represent scenarios with limited time for label collection. Of course LabelMe eventually reaches and surpasses the classification performance of HCGL, but requires time that may be infeasible in some scenarios.

Overall classification accuracy may be skewed by classes with higher distributions of pixels, but evaluating per-class classification accuracy shows that HCGL performs similarly or better than LabelMe for all classes but one. Fig. 4b shows per-class classification after 150 minutes of labeling (150 is selected to match the models used in Section IV). The *object* class is the least represented in the data and includes a variety of objects, e.g., light poles, traffic cones and cargo boxes. Low intra-class similarity caused few object samples to group together, and no *object* labels were collected by HCGL after 150 minutes. Although not shown due to space constraints, HCGL does eventually label object examples, but always yields lower accuracy than LabelMe for this class. Note that this was a difficult class for LabelMe as well. With a fully labeled training set (1,602 minutes), LabelMe achieves only 18% classification accuracy for the *object* class.

A labeled test set from environment A is not available, so we provide a qualitative pixel-wise classification comparison of HCGL and LabelMe. Fig. 5 shows six example test images, disjoint from the training set. We use LabelMe to create ground truth for these images, seen in the bottom row. Classifiers are trained using labeled data at the third markers from Fig. 3b. The selected examples show two instances where the classifiers perform similarly, an example where HCGL performs slightly worse than LabelMe (column three), and the last three columns are examples of HCGL's superior performance and illustrate the common mistakes made by the classifier trained using LabelMe. Specifically, the LabelMe classifier often misidentifies terrain further from the camera. This allows robots to make immediate decisions, but negatively impacts long term path planning. Qualitatively it can also be seen that HCGL commonly misclassifies *trees* and certain *objects* as *sky*, which are less costly for our navigation task. These mistakes occur because the *tree* and *object* classes are less represented than *sky* in the training set so fewer examples are collected by HCGL. However, the overall HCGL performance on these classes is still qualitatively high. Overall, HCGL collects significantly more label information even with 25% less human interaction time, and trains higher performing classifiers. A more detailed video of this qualitative comparison that supports our claims is provided as supplementary material.

IV. REAL-TIME NAVIGATION EXPERIMENTS

Pixel-wise accuracy quantitatively compares techniques on static data, but task-based evaluation judges perception relative to the end goal of successful navigation in outdoor environments. We compare several visual classifiers trained using labels collected by HCGL and LabelMe based on their ability to provide perception information to a real-time

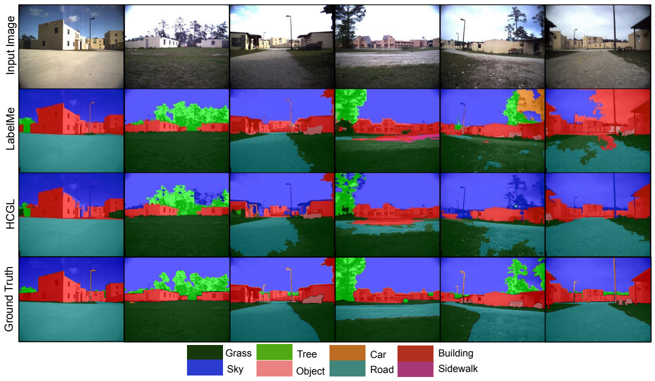
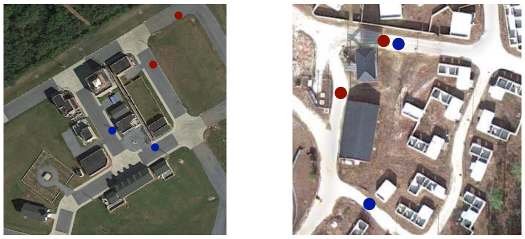


Fig. 5: Qualitative comparison between HCGL and LabelMe with a test set from Environment B.



(a) Environment A (b) Environment B

Fig. 6: Navigation waypoint maps for environments.

mapping and navigation framework.

A. Task Description

Our live navigation task requires a robot to use visual perception to plan paths between waypoints using specified terrain. These terrains are defined based on the composition of the road at testing locations. We use road traversal because roads are designed to provide navigation guidance to vehicles. For example, roads direct vehicles around buildings and hazards like bodies of water. Our experiments emulate these scenarios by defining waypoints (seen in Fig. 6) such that the most direct path to goals is not along a road.

Classifiers are compared based on successes and failures during multiple trials of the navigation task, where outcomes are defined as follows:

- **Success** - the robot autonomously traverses between waypoints using only road terrain without hitting objects
- **Success with Minor Errors** - the robot traverses between waypoints but either 1) traverses on non-road terrain for a short duration, or 2) requires operator intervention at least once but no more than twice for small adjustments in location or direction due to potential object collision or planner failure
- **Failure** - the robot cannot plan and execute a road traversal even with minimal operator intervention; visual perception has significant false-positive errors indicating no road path or constant planner updates result in no progress towards the goal

B. Hardware

The robot used in this work, the Clearpath Husky seen in Fig. 7, is a 39x26x14 inch wheeled platform, that is limited to a maximum velocity of 1 m/s. The Husky employs a MicroStrain 3DM-GX3-25 IMU, a Garmin 18 GPS and two Quad-Core Intel i7 Mini-ITX processing payloads, each with a 256 GB SSD running Ubuntu 14.04, ROS Indigo and experimental software. The Husky has a Velodyne HDL-32E LiDAR, which generates 360° point clouds at a range of 70 m and an accuracy of up to ±2 cm. Finally, the Husky collects image data using a Prosilica GT2750C, a 6 megapixel CCD color camera.

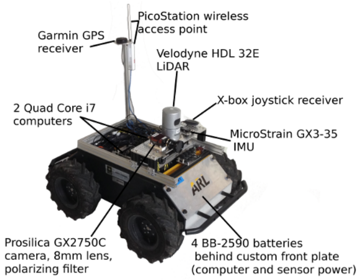


Fig. 7: Hardware configuration of the Clearpath Husky robot.

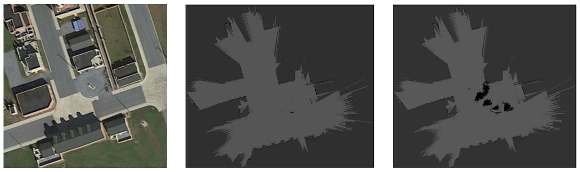
C. Mapping and Navigation

Our robot test platform employs a mapping and navigation system to enable accurate motion between desired waypoints. The mapping system, dubbed *OmniMapper*, consumes measurements from LiDAR for relative motion estimation and loop closure through ICP [26], GPS measurements [27] and camera images. A keyframe is created with each measurement as the robot moves through its environment; the robot's pose at this keyframe is optimized through *GTSAM* [28] to minimize residual error from all measurements.

A 2D local occupancy grid is created from each laser-scan keyframe through ray-tracing, where sufficient height above the ground is registered as an obstacle. When a new keyframe is added, or when a significant update is made to the map causing keyframe poses to change, the 2D occupancy grids are composited together into a negative log-odds grid, and thresholded into an obstacle map as in Fig. 8b.

A keyframe is also created for each classified image, and the pose of this record is updated with the mapping process such as with loop closures or GPS measurements. Whenever a new obstacle map is created, additional cells are marked as "obstacle" if those cells, when projected into classified images, overlap with pixels classified as one of the defined non-road terrains or an object class. In Fig. 8a, only *asphalt* and *concrete* make up the road for this testing location.

The corners of each map grid cell (10x10 cm) are projected into all classified images that observe that cell within a range of 7 meters. The classified images are rectified so the projected corners define a quad in the classified image. Each pixel in the projected quad has a label from the classifier and votes for that class to be applied to the ground cell. The ground cell is assigned the label with the highest number of votes. If this label does not represent road for navigation, the occupancy grid cell is given an obstacle value to prevent traversal through that cell. As seen in Fig. 8c, visual



(a) Environment (b) LiDAR map (c) Vision map

Fig. 8: Example obstacle maps for location two in environment A. Darker regions indicate obstacles and non-road terrain.

perception helps produce cost maps with specific terrain information, e.g., gravel regions are darker and avoided during path planning (discussed further in Section IV).

A kinematically feasible path is computed from the robot's current location to the goal location using the Search-Based Planning Library (SBPL) [29] using a set of motion primitives generated to match the Husky's kinematics. A smoothed local plan is chosen which follows the global plan closely while avoiding local obstacles not yet present in the global map. Planner failures occur if the occupancy grid prohibits an obstacle free path to the goal. This occurs in our experiments due to false-positive non-road classifications on road terrain. See [30] for more implementation details of the mapping and navigation systems used in this work.

D. Navigation Results - Environment A

Environment A is the primary location used for comparative evaluation since LabelMe was used to label its entire training set [24]. Four classifiers are trained and compared. We compare the labeling techniques given the same amount of labeling interaction time. **HCGL-150** and **LabelMe-150** represent classifiers trained after 150 minutes of labeling, which reflects scenarios where limited labeling time is available. This is just under one-tenth of the estimated total time (1,602 minutes) required to label the entire training set with LabelMe. To demonstrate results given no time restrictions, a classifier is trained using the entire training set, denoted as **LabelMe-1602**.

The final classifier is meant to show the benefits of using training data representing the most recent state of a robot's environment, and how HCGL easily facilitates the labeling of data upon arrival to a new or changed environment. We supplement the existing training set (collected several years ago) with 231 additional images collected during our experiments (disjoint from testing locations). Labeling was performed for 30 minutes with HCGL, and $\sim 27\%$ of the pixels in the new images were assigned labels. Without ground truth for this set, the amount of collected label noise is unknown. This set of labeled data is combined with the labeled data from HCGL-150 to train the final classifier, denoted as **HCGL-150+30**.

Navigation experiments are performed at two locations in the environment. Location one is illustrated with red waypoints in Fig. 6a, and roads are composed of *gravel*, *concrete* and *asphalt*. Thus, path planning must avoid grass terrain (the shortest path between waypoints) and several objects near the edge of the grass and road. Each trial represents a traversal from one waypoint to the other and are performed in both directions. Trials were run across multiple days and different times of day to capture performance under varying environment conditions. Table II compares the performance of each classifier at this first location.

TABLE II: Summary of navigation results for location one (red waypoints) in environment A.

	% Successes		
Label Model	No Errors	Minor Errors	% Failures
HCGL-150	0.500	0.000	0.500
LabelMe-150	0.333	0.167	0.500
LabelMe-1602	0.250	0.250	0.500
HCGL-150+30	**0.875**	**0.125**	0.000

HCGL-150 and LabelMe-150 perform similarly and inconsistently with a 50% failure rate. LabelMe-1602 exhibits the same failure rate, but also displays more minor errors during its successful trials. LabelMe-1602 uses the most labeled data to learn class boundaries with respect to the training set, but performs worse because the learned class boundaries changed. The classifiers trained after 150 minutes likely learned less definitive class boundaries making the environment changes less detrimental. Some observed changes from the training data include grass length, cloud coverage and illumination. HCGL-150+30 on the other hand, performs the navigation task very reliably because it represents a classifier that has adapted to the changed environment with new and additional training data. Minor errors involved the robot trying to plan a shortest path through the grass, entering the grass for a brief moment before backing out and successfully planning a road traversal route. These results demonstrate the positive impact of rapid label collection, even if a small fraction is noisy, when new training data is needed to adapt and improve visual perception.

Qualitative evaluation of visual perception shows the labeling models produce classifiers that make different mistakes. Fig. 9 includes examples explicitly chosen to depict some of the worst classified images by one or more models. HCGL-150 had many false-positive *concrete* classifications, which can be seen best in columns one, three and four. Columns three and five highlight that LabelMe-150 produced more false-positives of *object* and *building* classes on what was actually road terrain. LabelMe-1602 has cleaner results than the previous models, but also often misclassified *gravel* as *object* (seen in column three), and tended to misclassify *trees* as *buildings* (seen in columns one and two). Although still not perfect classification, HCGL-150+30 has the most accurate results compared to the ground truth, which yielded its superior navigation success and highlights the importance of being able to quickly collect large amounts of new labeled training data given environment changes.

The second location is depicted in Fig. 6a with blue waypoints. At this location, roads are composed of *concrete* and *asphalt*, whereas *gravel* terrain (shortest path between waypoints) is not road. Along the shortest road path are two objects (traffic cones) that the robot must also avoid. Terrain classification for classes with high inter-class similarity is

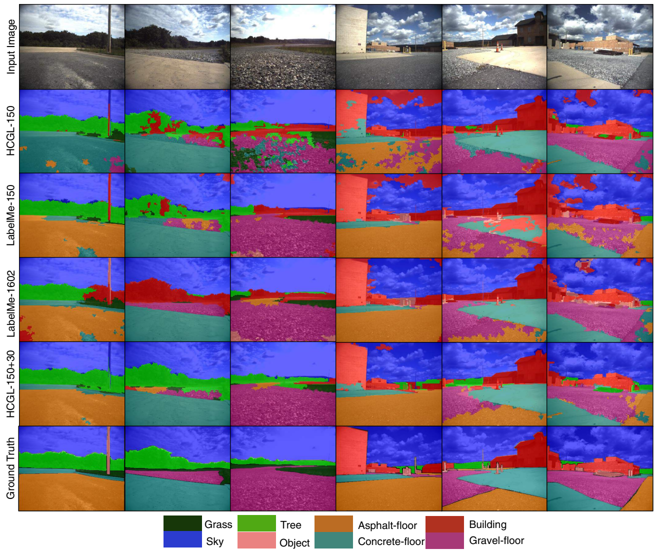


Fig. 9: Visual perception examples each labeling model. Images in the first three columns are from the first location (red waypoints), and the last three columns are images from the second location (blue waypoints).

TABLE III: Summary of navigation results for location two (blue waypoints) in environment A.

	% Successes		
Label Model	No Errors	Minor Errors	% Failures
LabelMe-1602	0.000	0.000	1.000
HCGL-150+30	**0.375**	**0.250**	0.375

important for successful traversal during this test.

Comparisons are made between LabelMe-1602 and HCGL-150+30; the most successful models at the first location in terms of successes and qualitative evaluation. Results are summarized in Table III and indicate that this navigation task is more challenging. However, HCGL-150+30 is still able to successfully navigate the majority of the time with only minor errors. Most failures and errors at this location were caused by classification confusion of *asphalt* and *gravel*. This can be seen in the last three columns of Fig. 9.

E. Navigation Results - Environment B

We use environment B, seen in Fig. 6b, to further test HCGL label collection in new domains. In this environment, roads are composed of a single terrain type labeled as *road*, and all other terrains and objects should be avoided during path planning. Training data for this environment was not available prior to the experiment so data was collected upon arrival. We chose to focus our navigation trial experiments on labels collected using HCGL to show the consistency of the system across multiple environments.

Due to space limitations, an in depth discussion and analysis of experiments in this environment is omitted, but an example trial can be seen in the supplementary material. Over 15 navigation trials were performed between both waypoint sets without any failure cases. Only minor path planning errors in a few trials caused the robot to traverse on the edge of the *grass* where it meets the *road*. These successes are used to confirm that small amounts of label noise collected by HCGL, in exchange for fast label collection, does not negatively impact path planning.

V. RELATED WORK

A. Visual Perception

Vision provides valuable perception for mobile robots. Terrain and obstacle classification are particularly important to help determine traversability. For example, visual terrain classification has been used to identify when legged robots should change gaits [6], [7], and aerial robots can identify possible landing sites or be used to communicate with ground robots when working in teams [5]. Visual perception is also being used for path planning on ground robots. Haselich et al. fuse 3D laser scans and camera images to perceive *road*, *rough* and *obstacle* terrain classes [1]. Haselich et al. is the first to mention the inability to adapt quickly to new environments due to the requirement of re-annotation.

Consequently, a significant amount of visual perception path planning research focuses on semi-supervised, self-supervised and on-line learning. Teleoperation has been used to define optimal routes to infer *path* and *non-path* labels for visual classifiers [31]. Ross et al. identify obstacles with an unsupervised, on-line technique that compares visual appearance and structure to learned environment models [9]. Roncancio et al. adapt a pre-trained supervised visual classifier on-line to identify *traversable* and *non-traversable* paths [8].

Other techniques pair vision with complimentary sensors. Visual features have been used to enhance RADAR *ground* prediction [3]. The correspondence between visual features and a robot's navigation experience, e.g., slippage, was used to identify *traversable* terrain [14]. Lookingbill et al. used a reverse optical flow technique to update visual classifiers with the appearance of obstacles beyond the range of stereo vision [16]. Other self-supervised learning examples include combining vision and LiDAR [15], [17].

These examples adapt terrain classifiers without the time consuming labeling process. However, the lack of human supervision has limited most of this work to binary classification, e.g., *traversability*. Unfortunately, these approaches do not extend to more complex multi-class tasks such as verbal navigation commands from human to robot [18].

B. Label Collection

Techniques to reduce labeling effort for multi-concept classification have emerged in the vision domain. Active learning frameworks [32], [33] identify and label a diverse subset of training data with iterative human interaction and supervised classifier re-training. Incremental and active clustering [21], [22] are iterative group labeling approaches where a single label is assigned to multiple images simultaneously. However, the active or on-line re-training and re-clustering creates latency between label assignments, and is expensive for real-world environment adaptation.

Related, there has been work on how to reduce labeling effort for video data. Xie et al. introduce a label transfer approach where coarse 3D annotations of street scenes can be transfered to 2D images [34]. Other semi-supervised label propagation for video streams has also been achieved with random forests [35] and a mixture of temporal trees [36]. These approaches use the information encoded by temporal consistency to reduce labeling effort, but are not compatible for large sets of non-sequential training images, e.g., environment A in our experiments.

VI. Conclusion

Real-time visual perception for mobile robots is only as useful as its ability to quickly adapt to changing environments. This paper modified an efficient label collection technique, called Hierarchical Cluster Guided Labeling (HCGL), for multi-concept environment data. It was shown that while HCGL trades some label accuracy for reduced adaptation latency, this label noise does not significantly impact visual perception for navigation. Using this technique, high quality visual perception can be obtained in new environments with only a few hours of labeling effort from a human annotator.

The multi-concept semantics provided by HCGL allow this work to generalize to more complex variations of path planning tasks. This includes assigning variable costs to terrains based on robot capabilities, and path planning with verbal navigation cues given during human-robot interaction. Future work also includes augmenting HCGL to be even more effective through on-line label collection and adaptation.

References

[1] M. Häselich, M. Arends, N. Wojke, F. Neuhaus, and D. Paulus, "Probabilistic terrain classification in unstructured environments," *Robotics and Autonomous Systems*, vol. 61, no. 10, pp. 1051–1059, 2013.

[2] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "Autonomous robot navigation in highly populated pedestrian zones," *Journal of Field Robotics*, vol. 32, no. 4, pp. 565–589, 2015.

[3] A. Milella, G. Reina, and J. Underwood, "A self-learning framework for statistical ground classification using RADAR and monocular vision," *Journal of Field Robotics*, vol. 32, no. 1, pp. 20–41, 2015.

[4] S. Manjanna, G. Dudek, and P. Giguere, "Using gait change for terrain sensing by robots," in *Int. Conf. on Computer and Robot Vision*, 2013, pp. 16–22.

[5] Y. N. Khan, A. Masselli, and A. Zell, "Visual terrain classification by flying robots," in *Int. Conf. on Robotics and Automation*, 2012, pp. 498–503.

[6] P. Filitchkin and K. Byl, "Feature-based terrain classification for littledog," in *Int. Conf. on Intelligent Robots and Systems*, 2012, pp. 1387–1392.

[7] S. Zenker, E. E. Aksoy, D. Goldschmidt, F. Worgotter, and P. Manoonpong, "Visual terrain classification for selecting energy efficient gaits of a hexapod robot," in *Int. Conf. on Advanced Intelligent Mechatronics*, 2013, pp. 577–584.

[8] H. Roncancio, M. Becker, A. Broggi, and S. Cattani, "Traversability analysis using terrain mapping and online-trained terrain type classifier," in *Intelligent Vehicles Symposium*, 2014, pp. 1239–1244.

[9] P. Ross, A. English, D. Ball, B. Upcroft, and P. Corke, "Online novelty-based visual obstacle detection for field robotics," in *Int. Conf. on Robotics and Automation*, 2015, pp. 3935–3940.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer Vision and Pattern Recognition*, June 2015.

[12] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

[13] D. Munoz, "Inference machines: Parsing scenes via iterated predictions," Ph.D. dissertation, The Robotics Institute, Carnegie Mellon University, June 2013.

[14] D. Kim, J. Sun, S. M. Oh, J. M. Rehg, and A. F. Bobick, "Traversability classification using unsupervised on-line visual learning for outdoor robot navigation," in *Int. Conf. on Robotics and Automation*, 2006, pp. 518–525.

[15] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski, "Self-supervised monocular road detection in desert terrain." in *Robotics: Science and Systems*, 2006.

[16] A. Lookingbill, J. Rogers, D. Lieb, J. Curry, and S. Thrun, "Reverse optical flow for self-supervised adaptive autonomous robot navigation," *Int. Journal of Computer Vision*, vol. 74, no. 3, 2007.

[17] S. Zhou, J. Xi, M. W. McDaniel, T. Nishihata, P. Salesses, and K. Iagnemma, "Self-supervised learning to visually detect terrain surfaces for autonomous robots operating in forested terrain," *Journal of Field Robotics*, vol. 29, no. 2, pp. 277–297, 2012.

[18] D. Summers-Stay, T. Cassidy, and C. R. Voss, "Joint navigation in commander/robot teams: Dialog & task performance when vision is bandwidth-limited," *V&L Net*, p. 9, 2014.

[19] M. Wigness, B. A. Draper, and J. R. Beveridge, "Efficient label collection for unlabeled image datasets," in *Computer Vision and Pattern Recognition*, 2015, pp. 4594–4602.

[20] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: a database and web-based tool for image annotation," *Int. Journal of Computer Vision*, vol. 77, no. 1-3, 2008.

[21] C. Galleguillos, B. McFee, and G. Lanckriet, "Iterative category discovery via multiple kernel metric learning," *Int. Journal of Computer Vision*, vol. 108, no. 1-2, pp. 115–132, 2014.

[22] C. Xiong, D. M. Johnson, and J. J. Corso, "Spectral active clustering via purification of the k-nearest neighbor graph," in *European Conference on Data Mining*, 2012.

[23] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[24] C. Lennon, B. Bodt, M. Childers, R. Camden, A. Suppé, L. Navarro-Serment, and N. Florea, "Performance evaluation of a semantic perception classifier," Army Research Labs, Tech. Rep. ARL-TR-6653, 2013.

[25] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. Journal of Computer Vision*, vol. 59, no. 2, 2004.

[26] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems*, vol. 2, no. 4, 2009.

[27] J. G. Rogers, J. R. Fink, E. Stump, *et al.*, "Mapping with a ground robot in GPS denied and degraded environments," in *American Control Conference*, 2014, pp. 1880–1885.

[28] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *Int. Journal of Robotics Research*, vol. 25, no. 12, 2006.

[29] B. J. Cohen, S. Chitta, and M. Likhachev, "Search-based planning for manipulation with motion primitives," in *Int. Conf. on Robotics and Automation*, 2010, pp. 2902–2908.

[30] J. Gregory, J. Fink, E. Stump, J. Twigg, J. Rogers, D. Baran, N. Fung, and S. Young, "Application of multi-robot systems to disaster-relief scenarios with limited communication," in *Field and Service Robotics*, 2015.

[31] K. Konolige, M. Agrawal, R. C. Bolles, C. Cowan, M. Fischler, and B. Gerkey, "Outdoor mapping and navigation using stereo vision," in *Experimental Robotics*. Springer, 2008, pp. 179–190.

[32] X. Li and Y. Guo, "Adaptive active learning for image classification," in *Computer Vision and Pattern Recognition*, 2013.

[33] B. Siddiquie and A. Gupta, "Beyond active noun tagging: Modeling contextual interactions for multi-class active learning," in *Computer Vision and Pattern Recognition*, 2010, pp. 2979–2986.

[34] J. Xie, M. Kiefel, M. Sun, and A. Geiger, "Semantic instance annotation of street scenes by 3d to 2d label transfer," in *Computer Vision and Pattern Recognition*, 2016.

[35] N. S. Nagaraja, P. Ochs, K. Liu, and T. Brox, "Hierarchy of localized random forests for video annotation," in *Joint DAGM and OAGM Symposium*. Springer, 2012, pp. 21–30.

[36] V. Badrinarayanan, I. Budvytis, and R. Cipolla, "Mixture of trees probabilistic graphical model for video segmentation," *Int. Journal of Computer Vision*, vol. 110, no. 1, pp. 14–29, 2014.